Titre: Title:	Segmentation and tracking of cavity area in thoracoscopic video sequences
Auteur: Author:	Yue Yun Shu
Date:	2005
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Shu, Y. Y. (2005). Segmentation and tracking of cavity area in thoracoscopic video sequences [Master's thesis, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/7682/

UNIVERSITÉ DE MONTRÉAL

SEGMENTATION AND TRACKING OF CAVITY AREA IN THORACOSCOPIC VIDEO SEQUENCES

YUE YUN SHU
DÉPARTEMENT DE GÉNIE INFORMATIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE INFORMATIQUE)
JUIN 2005

Your file Votre référence
ISBN: 978-0-494-16855-4
Our file Notre référence
ISBN: 978-0-494-16855-4

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé:

SEGMENTATION AND TRACKING OF CAVITY AREA IN THORACOSCOPIC VIDEO SEQUENCES

présenté par: SHU Yue Yun
en vue de l'obtention du diplôme de: Maîtrise ès sciences appliquées
a été dûment accepté par le jury d'examen constitué de:

Mme NICOLESCU Gabriela, Doct., présidente
M. BILODEAU Guillaume-Alexandre, Ph.D., membre et directeur de recherche
Mme CHERIET Farida, Ph.D., membre et codirecteur de recherche
M. OZELL Benoit, Ph.D., membre

ACKNOWLEDGEMENTS

Guillaume-Alexandre Bilodeau

Farida Cheriet

Luke Windisch

Qing Hu Liao

Et le jury.

RÉSUMÉ

La chirurgie minimalement invasive est un procédé qui consiste à insérer des instruments et un endoscope à travers de petites incisions sur le tronc du patient. L'endoscope acquiert des images des organes internes et des instruments, et le chirurgien effectue l'opération en visualisant les images sur un moniteur. Les avantages de ce genre de procédé sont la réduction des inconvénients aux patients, de meilleurs diagnostics et meilleurs résultats thérapeutiques. Comparée aux chirurgies conventionnelles, cette méthode réduit les risques aux patient et les coûts d'hospitalisation.

La discectomie thoracique est de cette famille de procédé. Elle consiste à enlever des disques intervertébraux de patients souffrant de la scoliose. Ce procédé est appliqué pour fusionner quelques niveaux vertébraux afin d'arrêter la progression des déformations scoliotiques. En utilisant un endoscope, le chirurgien doit s'orienter à partir d'images endoscopiques 2D. Cela demande plus d'habiletés que la chirurgie invasive traditionnelle, parce que les images acquises sont détériorées par le bruit, l'illumination est pauvre, le champ visuel est limité; et le plus important, la perception de profondeur est perdue. Sans perception de profondeur, il est difficile d'évaluer la distance entre l'instrument et la moelle épinière avec précision, ce qui est très important lors de l'opération.

Pour aider le chirurgien, nous devons reconstruire en 3D la région opérée à partir des images 2D endoscopiques pour permettre le rétablissement de la perception de profondeur. La première étape de la reconstruction 3D est de segmenter la région opérée dans les images. Dans ce travail, nous nous concentrons sur la localisation de la frontière du disque enlevé, identifiée dans l'image comme une cavité. La région segmentée correspondant à la cavité aidera la reconstruction en 3D afin de permettre l'estimation de la distance relative entre l'instrument et la moelle épinière. Les objectifs spécifiques de ce projet sont:

- Segmentation de la cavité afin de la distinguer des autres tissus et instruments;
- Suivi de la cavité pour la localiser d'image en image dans les vidéos.

Le premier objectif a été accompli au moyen de prétraitements sur l'image, une segmentation par graphe suivie d'une segmentation par fusion de régions en plusieurs étapes. Le prétraitement inclut un lissage gaussien, un ajustement de la luminosité et du contraste, la conversion de l'espace de couleur et l'élimination de réflexions spéculaires. Ensuite, une segmentation par graphe, qui est une variante efficace de l'algorithme récursif de l'arbre de recouvrement minimum (RSST), est appliquée sur l'image prétraitée pour obtenir une segmentation relativement grossière. Ensuite, la carte de segmentation est traitée par la segmentation par régions en plusieurs étapes. L'algorithme de segmentation par graphe emploie la similitude entre les niveaux de gris comme critère pour grouper les pixels en des régions homogènes, tandis que la segmentation par régions en plusieurs étapes utilise différents critères aux différentes étapes pour fusionner les régions en des structures ou objets plus significatifs. Après avoir appliqué notre méthode originale de segmentation sur une image, la région de la cavité est bien séparée des autres régions. L'identification de la frontière de la cavité à ce stade est de beaucoup simplifiée. La frontière de la cavité sera employée en tant que contour initial pour l'algorithme de suivi par "snakes".

Le deuxième objectif est réalisé au moyen d'une combinaison de "snake" traditionnel et de "snake" GVF (Gradient Vector Flow). Le "snake" traditionnel peut réguler le contour et le déplacer aux arêtes les plus près, alors que le "snake" GVF peut déplacer le contour aux arêtes fortes présentes dans un plus grand voisinage. Au début du processus de suivi, le même prétraitement est appliqué sur l'image, et le flux des vecteurs de gradient (GVF) est calculé pour l'image prétraitée. Le contour de l'image précédente ou indiquée par l'utilisateur est alors déformé vers la nouvelle frontière dans l'image courante par un processus de réduction d'énergie du "snake" traditionnel. Par la suite, la nouvelle frontière est déformée encore avec le "snake" GVF. Le résultat du "snake" GVF est interpolé pour former un contour dense, qui est considéré comme contour initial pour la prochaine image. Par conséquent le contour original est suivi d'image en image. Nous avons également conçu une logique de perte de suivi pour permettre une remise en

marche facile de la segmentation et du suivi en cas de perte du suivi due à un changement de scène ou d'autres événements spéciaux.

Les résultats obtenus montrent que nous avons atteint nos objectifs puisque notre méthode de fusion par plusieurs critères peut améliorer les résultats de segmentation de manière significative. De plus, le processus intégré de segmentation peut segmenter la cavité adéquatement pour une variété d'images difficiles. Ils montrent également que le "snake" traditionnel combiné au "snake" GVF peut efficacement faire le suivi de la cavité pendant plusieurs images.

ABSTRACT

The minimally invasive surgery is a procedure in which instruments and an endoscope are inserted through small incisions on the patient's body. The endoscope acquires images of internal organs and of operational instruments; and the surgeon performs the operation by viewing the images displayed on a video monitor. The advantages of this kind of procedure are minimal inconvenience to patients, and improved diagnostic accuracy and therapeutic outcome. Compared to traditional invasive surgeries, this method reduces patient risk and costly in-hospital recovery periods.

Thoracic discectomy is that kind of procedure to remove inter-vertebral disc from the spine of patients suffering from scoliosis. This procedure is performed to fuse some vertebral levels in order to stop the progression of spinal deformities. By using an endoscope, the surgeon has to decide his action by looking at the 2D endoscopic images during thoracic discectomy. That is actually more difficult than traditional surgery, because the acquired images are deteriorated by noises, poor illumination and limited field of view; and more importantly, the depth perception is lost. Without depth perception, it is hard to tell the distance between the operational instrument and the target organ precisely, which is very important during operation.

To help the surgeon, we need to reconstruct 3D images of the operational area from the 2D endoscopic video sequence to allow recovery of depth perception. The first step of 3D reconstruction is to segment the operational areas in images. In this work, we focus on localizing the boundary of the removed disc, identified in the image as a cavity. The segmented cavity areas will then be used for 3D reconstruction of the cavity, which will help estimate the relative distance between the instrument and the spinal cord. The specific objectives of this project are:

- Segmentation of the cavity from the other tissues and instruments;
- Tracking of the cavity to precisely locate the cavity area from frame to frame.

The first subtask was accomplished by means of image pre-processing and graph-based segmentation followed by multistage region merging. The pre-processing includes

Gaussian smoothing, brightness and contrast enhancement, colour space conversion and specular reflections removal. Then a graph-based segmentation, which is an efficient variant of the recursive shortest spanning tree (RSST) algorithm, is applied on the pre-processed image to obtain a relatively coarse segmentation. After that, the segmentation map is treated by a multistage region merging algorithm. The graph-based segmentation algorithm uses similarity on grey-level as the criterion to group pixels into homogeneous regions, whereas the region merging algorithm uses different criteria at different stages to merge regions into more meaningful structures or objects. After running our novel segmentation method on a frame, the cavity area is well separated from other regions. Identifying the boundary of the cavity at this stage is much simplified. The identified boundary of the cavity will be used as initial contour of the tracking algorithm.

The second subtask was accomplished by use of a combination of traditional snake and gradient vector flow (GVF) snake. Traditional snake is able to regulate the contour and move it to the edges in close vicinity, while GVF snake can move the contour to strong edges in a larger range. At the beginning of tracking process, the same pre-processing is applied on the current frame, and the gradient vector flow is computed from the pre-processed image. The contour from previous frame or specified by the user is then evolved to the new boundary in current frame through the energy minimizing process of traditional snake. Next, the new boundary is deformed again in the guidance of GVF snake. The result of GVF snake is interpolated to form a dense contour, which is used as the initial contour of next frame. Therefore the original contour is tracked from frame to frame as the process continues. We have also designed a lost tracking logic to allow an easy restarting of the segmentation and tracking process in case of tracking lost due to scene cut or other special events.

Results obtained show that we have reached our objectives as our multi-criteria merging method can improve segmentation results significantly and the integrated segmentation process can segment the cavity area adequately for a variety of challenging images. They also show that the combined traditional snake and GVF snake can effectively track the cavity in a number of frames.

CONDENSÉ EN FRANÇAIS

Introduction

L'endoscopie est un procédé chirurgical minimalement invasif qui utilise de multiples petites incisions sur le corps du patient à travers lesquelles le chirurgien passe des outils et une caméra pour conduire une opération. Certaines recherches ont été conduites ces dernières années pour le traitement des images chirurgicales endoscopiques afin d'améliorer la qualité des opérations. La plupart des chercheurs se sont concentrés sur la segmentation des instruments dans les images et le suivi de leurs mouvements.

Dans notre cas, nous nous sommes intéressés aux images d'un procédé chirurgical différent avec des propriétés différentes. Les images que nous traitons proviennent d'une procédure de discectomie thoracique. La discectomie thoracique utilise la technique de l'endoscopie pour enlever des disques intervertébraux de patients souffrant de la scoliose. Une des difficultés de ce genre de chirurgie est que la perception de profondeur est perdue dans les 2D images acquises. Sans perception de profondeur, il est difficile de déterminer la distance entre l'instrument et la moelle épinière avec précision. Pour aider le chirurgien, nous devons reconstruire en 3D la région opérée obtenue par l'endoscope afin de récupérer la perception de profondeur. La première étape de la reconstruction 3D est de segmenter les régions contenues dans une série d'images. Dans ce travail, nous nous concentrons sur la localisation de la frontière du disque enlevé, identifiée dans l'image comme cavité. Nos objectifs sont la segmentation de la cavité pour la distinguer des autres tissus et des instruments et le suivi de la cavité segmentée d'image en image.

À la différence des instruments chirurgicaux ou de tout autre objet usuel, la cavité à identifier dans les images endoscopiques est non structurée et a une forme qui change d'image en image en fonction de la région précise observée par l'endoscope. La forme variable de la cavité nous mène naturellement aux modèles déformables. Puisque les modèles déformables peuvent représenter des formes complexes, ils ont trouvé des applications en traitement d'images médicales. Des modèles déformables sont souvent employés dans le suivi de contours, dans lequel le contour est déformé sous l'influence de

forces internes et de forces externes (de l'image). Le "snake" est une classe spéciale des modèles déformables. Habituellement, le contour est initialisé par l'interaction avec un utilisateur, et ensuite il est déplacé dans une autre image pour s'adapter aux caractéristiques de celle-ci, par exemple, le gradient de l'image. Sur la base du modèle du "snake", nous avons conçu une méthode incorporant la segmentation et le suivi pour atteindre nos objectifs. Le scénario typique de notre application est décrit comme suit. D'abord, une vidéo est chargée et une image de départ de celle-ci est choisie. Puis le système effectue une segmentation sur cette image qui devrait clairement distinguer la cavité des autres tissus. Ensuite, la région de la cavité est simplement indiquée par un clic de souris dans la carte de segmentation, et alors le système traitera automatiquement les autres images de la vidéo pour faire le suivi de la région spécifiée.

Dans cette thèse, nous présentons un algorithme de segmentation qui intègre la segmentation par graphe et la segmentation à plusieurs étapes par fusion de régions. Aussi nous présentons une application des modèles déformables pour le suivi d'une région par une combinaison de "snake" traditionnel et de "snake" GVF. Notre contribution spécifique est le processus de fusion de régions en plusieurs étapes suivant une segmentation grossière par graphe pour surmonter les inconvénients de la segmentation par graphe. De plus, nous proposons un nouveau modèle déformable qui combine le "snake" traditionnel et le "snake" de GVF qui peut conserver une grande région de capture sans souffrir de bruits de propagation pendant le processus de suivi.

1 État de l'art

1.1 Méthodes de segmentation d'images

Le but de la segmentation d'image est de décomposer une image en régions sans recouvrement qui sont significatives pour une application particulière. On a proposé de nombreuses méthodes de segmentation dans la littérature, nous résumons sept méthodes de base comme suit.

1) Seuillage d'histogrammes. Le seuillage est peut-être la forme la plus intuitive de segmentation d'image. Une image peut être simplement segmentée par des seuils

sur les valeurs d'intensité des pixels. Dans le seuillage basé sur un histogramme, l'histogramme d'image est utilisé pour placer divers seuils pour diviser l'image donnée en des régions distinctes. Le seuillage est simple et rapide, mais dans des applications complexes telles que le traitement d'image médicale, la sélection des seuils appropriés est souvent très difficile ou même impossible.

2) Segmentation basée sur les arêtes. Les arêtes sont des pixels où une fonction d'image (par exemple l'intensité) change abruptement. Un changement de la fonction d'image peut être décrit par un gradient qui pointe dans la direction de la plus grande croissance. Le gradient est approximé dans les images numériques par une convolution avec les opérateurs de détection d'arêtes [8, 16]. La segmentation basée sur les arêtes se fondent sur les arêtes trouvées dans une image par ces opérateurs - ces endroits dans l'image marquent des discontinuités dans le niveau de gris, la couleur, la texture, etc. L'image résultant de la détection d'arêtes ne peut pas être utilisée comme résultat de segmentation. Des étapes de transformations supplémentaires doivent suivre pour combiner les arêtes dans des chaînes qui correspondent mieux aux frontières dans l'image.
3) Segmentation basée sur les régions. Les méthodes de segmentation basées sur les régions sont généralement meilleures dans des images bruitées où il est difficile de détecter les arêtes, parce que des pixels bruités provoquant de fausses arêtes peuvent être intégrés dans des régions suivant les propriétés des pixels voisins. Des algorithmes de croissance de régions sont basés sur la croissance de régions homogènes selon certaines caractéristiques comme l'intensité, la couleur ou la texture. La séparation de régions est l'opposé du fusionnement de régions. Elle commence par l'image entière représentée comme une seule région. Les régions existantes de l'image sont séquentiellement séparées pour satisfaire les critères donnés d'homogénéité. Une combinaison de la séparation et de la fusion de régions peut produire une méthode avec les avantages des deux approches. La théorie des graphes est fréquemment utilisée dans les méthodes basées sur les régions. Elles considèrent chaque pixel comme un noeud dans un graphe. Un arc

joint les pixels voisins dont les propriétés sont assez semblables. Une méthode efficace de segmentation par graphe est adoptée dans ce travail, et les détails sont donnés à la section 1.4.

4) Segmentation par agrégation. Une vue naturelle pour la segmentation est de déterminer quels composants d'un ensemble de données vont "ensemble". C'est un problème connu sous le nom d'agrégation (clustering). Il y a deux algorithmes de base pour l'agrégation. Dans l'agrégation séparative, l'ensemble de données complet est considéré comme un agrégat, et des agrégats sont récursivement divisés pour donner de meilleurs groupes. Dans l'agrégation agglomérative, chaque donnée élémentaire est considérée comme un agrégat, les agrégats sont récursivement fusionnés pour donner de meilleurs groupes. Certaines difficultés de l'agrégation inclus la détermination du nombre d'agrégats et leur validité. Les algorithmes d'agrégation n'incorporent pas directement la modélisation spatiale et peuvent donc être sensibles à l'inhomogénéité d'intensité et au bruit.

5) Segmentation par la morphologie mathématique. La morphologie mathématique utilise des opérations sur les ensembles pour extraire des composants d'image. La plupart des opérations morphologiques sont basées sur des opérations d'extension (dilatation) et de rétrécissement (d'érosion). L'outil morphologique de base pour la segmentation d'image est le "watershed" [24]. La segmentation morphologique se compose essentiellement de deux étapes. La première, connu comme l'extraction de marqueur vise à identifier un ensemble de zones homogènes initiales. La deuxième étape de décision est basée sur le processus de "watershed", qui assigne les pixels restants aux zones appropriées. La segmentation par "watershed" produit des images sévèrement sur-segmentées avec des centaines ou des milliers de régions. D'autres traitements doivent être appliqués pour produire un résultat approprié. Des opérateurs morphologiques sont utilisés dans notre méthode de segmentation par région en plusieurs étapes pour trouver les régions adjacentes à une région spécifique.

6) Segmentation par des méthodes statistiques. Les méthodes statistiques de segmentation utilisent les techniques développées dans le domaine de la reconnaissance statistique de patrons pour classifier des pixels dans différentes régions. Le système de reconnaissance fonctionne en deux modes. En mode d'entraînement, le module d'extraction de caractéristiques trouve les caractéristiques appropriées pour représenter les patrons d'entrée, et le classificateur est entraîné pour diviser l'espace des caractéristiques. En mode de classification, le classificateur assigne le patron d'entrée à une des classes de patrons sur la base des caractéristiques mesurées. Un inconvénient des méthodes statistiques est qu'elles sont des méthodes fondamentalement supervisées puisqu'elles exigent les données d'entraînement qui souvent sont des segmentations manuelles. En outre, comme les méthodes d'agrégation, elles ne font généralement pas de modélisation spatiale.
7) Segmentation par appariement de patrons. Parfois, le but de la segmentation est de détecter un objet spécifique dans l'image. Si un patron décrivant un objet spécifique est disponible, la détection d'objet devient un processus d'appariement des caractéristiques d'un patron avec ceux de l'image sous l'analyse. La détection d'objets avec un appariement exacte demande généralement beaucoup de calculs et la qualité de l'appariement dépend des détails et du degré de précision fourni par le patron d'objet. En effet, un de nos objectifs est de détecter la région de la cavité dans des images endoscopiques, mais malheureusement, dans notre application, un patron fixe correspondant à la cavité ne peut pas être formulé puisque ces formes changent considérablement, et sa couleur et sa texture changent selon les incisions et la propagation du sang.

1.2 Méthodes de suivi d'objets

Des objets mobiles sont toujours associés aux vidéos. Dans une vidéo, des images sont prises à intervalles temporels très rapprochés. Par conséquent, les contenus de deux images consécutives sont habituellement étroitement liées. Les caractéristiques

temporelles dans les images s'appellent habituellement l'information de mouvement. Le suivi d'objets dans les vidéos implique de vérifier la présence d'un objet et de surveiller ses changements spatiaux et temporels dans différentes images. Il y a généralement deux manières de faire le suivi d'objets:

1) Segmentation en utilisant l'information de mouvement. En calculant l'information de mouvement, il est possible de différencier les régions mobiles dans une image par rapport à l'arrière-plan, particulièrement quand des données mobiles saillantes sont capturées. La base de la segmentation temporelle est l'information de mouvement. Deux modèles sont plus largement adoptés pour la description d'information de mouvement, à savoir le modèle global de mouvement et le modèle flux optique. L'information de mouvement peut être utilisée indépendamment ou conjointement avec d'autres caractéristiques de l'image dans le procédé de segmentation. Cependant, le mouvement est seulement fiable pour les objets texturés, ce qui n'est pas le cas pour la cavité dans notre application.
2) Suivi par les modèles déformables. Les modèles déformables sont appropriés aux cas où les objets changent en raison de déformations rigides et non rigides. Dans cette approche, un contour de l'objet à l'étude est d'abord identifié, par exemple par un algorithme de segmentation d'image. Une transformation sur le contour prototype est appliquée pour le déformer afin qu'il s'adapte aux arêtes saillantes de l'image d'entrée. Une fonction objective avec des paramètres de transformation qui modifient la forme du contour est formulée en reflétant le coût de telles transformations. La fonction objective est minimisée en mettant à jour itérativement les paramètres de transformation pour correspondre le mieux possible à l'objet [40]. En raison de la nature déformable de la cavité dans les images endoscopiques, les modèles déformables sont les plus attrayants pour le suivi. Dans ce travail, nous avons utilisé des "snakes" pour faire le suivi de la région de la cavité dans les images endoscopiques.

1.3 Applications au traitement d'images médicales

L'imagerie médicale jouant un rôle de plus en plus important dans le diagnostic et le traitement de la maladie, des techniques de segmentation ont été appliquées pour extraire des informations cliniques utiles sur les structures anatomiques obtenues par des modalités telles que le rayon X, CT (Computer Tomography), MRI (Magnetic Resonance Imaging), PET (Position Emission Tomography) et ultrasons [47]. Une observation commune au sujet des méthodes de segmentation utilisées est qu'une seule méthode accomplie rarement la tâche. Il est souvent nécessaire de combiner plusieurs techniques de segmentation de différentes catégories pour obtenir un résultat utile. Les modèles déformables ont attiré une attention particulière en imagerie médicale. Les modèles 3D et 2D déformables ont été utilisés pour segmenter, visualiser, faire le suivi et mesurer une variété de structures anatomiques du macroscopique au microscopique [4]. Bien qu'il y ait de nombreuses applications en imagerie médicale dans la littérature, peu portent sur des images endoscopiques chirurgicales [1, 2, 32, 33].

2 Segmentation d'images endoscopiques

2.1 Sélection des caractéristiques

Beaucoup de caractéristiques d'images peuvent être utilisés comme critères d'homogénéité pour la segmentation d'images, telle que la couleur, l'intensité (niveau de gris), la texture, les vecteurs de mouvement, les modèles élémentaires, etc. Parmi ces caractéristiques des images, l'intensité est considérée comme la plus appropriée comme critère de segmentation. Puisqu'il y a peu de texture, et aucun arrière-plan ou premier plan dominant dans des images endoscopiques, l'information de texture et de mouvement n'est pas fiable dans cette situation. D'autre part, les différences de couleur entre les régions sont petites, et la forme des régions est variable. Ainsi les modèles de couleur et de forme ne sont pas des critères fiables non plus. Par conséquent, l'intensité est choisie en tant que critère de base pendant la segmentation.

2.2 Prétraitement des images

Les images sont fondamentalement un ensemble de pixels qui sont souvent moins qu'une représentation parfaite de réalité. Pour des images endoscopiques chirurgicales, en raison du bruit, de l'illumination faible et de la similitude entre les couleurs, les frontières entre différentes structures ne sont pas claires. Il y a également quelques réflexions spéculaires non désirées dans les images. Ainsi nous devons faire certains prétraitements pour rendre la frontière des régions plus claires avant d'appliquer l'algorithme de segmentation. Quatre procédures sont utilisées pour prétraiter les données: Lissage gaussien, ajustement de la luminosité et du contraste, conversion des couleurs en intensité, et suppression des réflexions spéculaires. Le lissage gaussien nous permet d'enlever des variations et le bruit. Nous utilisons un opérateur de convolution 2D avec un masque 5 x 5. Ajuster la luminosité et le contraste nous permet d'améliorer la visibilité de l'information. Nous utilisons l'algorithme classique d'ajustement de luminosité/contraste. L'image de couleur est alors convertie en image d'intensité pour la segmentation. Il y a beaucoup de manières de faire la conversion de l'espace de couleur. Nous avons choisi d'utiliser la norme de Rec. 709 [55] parce qu'elle donne moins de poids au composant rouge, et par conséquent augmente les différences entre les régions rougeâtres. Finalement, le prétraitement se termine en enlevant les faux pixels qui sont en fait des pixels d'intensité très élevée causés par des réflexions spéculaires sur les instruments polis et les liquides dans la scène. En utilisant un simple seuillage par histogramme, l'intensité de ces pixels est tronquée. C'est très important, parce que ces pixels ont de grands poids et ils peuvent affecter la croissance de régions inopinément.

2.3 Segmentation par graphe

Notre méthode est basée sur une variante efficace de l'algorithme RSST [5]. Dans RSST, une image 2D est décrite comme un graphe non orienté $G = (V, E)$, dans lequel les pixels $v_i \in V$ correspondent aux noeuds dans le graphe, et les liens des pixels voisins forment les arcs $e_k = (v_i, v_j) \in E$. Chaque noeud et chaque arc a un poids correspondant. Le poids

d'un noeud est l'intensité (niveau de gris) dans notre cas, et le poids d'un arc est la différence de poids aux deux extrémités. Pendant le processus de segmentation, des pixels voisins semblables sont groupés en des régions, et le poids d'une région est alors simplement l'intensité moyenne des pixels composants celle-ci. Une segmentation est une partition de V en régions tels que chaque région correspond à un sous-ensemble connecté d'arcs de E. La segmentation est réalisée récursivement en trouvant l'arc avec le poids le plus faible et en fusionnant les deux régions (qui sont à l'origine des pixels) reliées par l'arc. Après chaque fusion, les poids de régions affectées sont recalculés. L'algorithme traditionnel de segmentation par RSST [58] recalcule les poids des arcs et puis trouve l'arc avec le poids le plus faible pour le fusionner. Puisque le tri des arcs est lent car il y a beaucoup d'arcs (par exemple 153 040 arcs dans une image typique de 320 x 240), le mettre dans le cycle de calcul prend beaucoup de temps. Un algorithme plus efficace [5] ne recalcule pas les poids et ne fait pas le tri des arcs, mais utilise un critère interne de différence pour juger si deux régions peuvent être fusionnées. Bien que l'algorithme efficace complique la sélection des critères d'arrêt et de segmentation, il est beaucoup plus rapide (fonctionnement presque en temps réel sur un PC Pentium IV 2.4GHz). Nous utilisons cet algorithme efficace pour produire une première carte de segmentation comme entrée à notre méthode de segmentation par fusion en plusieurs étapes.

2.4 Fusion de régions en plusieurs étapes

Bien que l'algorithme de segmentation par graphe, tel RSST soit une approche relativement bonne, elle ne produit pas toujours des résultats acceptables. C'est parce que les méthodes par graphes ont quelques inconvénients intrinsèques. Nous avons trouvé expérimentalement qu'une segmentation fine peut fusionner la région de la cavité avec d'autres parties dans des images endoscopiques, mais que nous pouvons employer le résultat d'une segmentation grossière dans notre nouvelle méthode qui utilise de l'information spécifique à l'application, et surmonter ainsi les inconvénients des méthodes de segmentation par graphes. Notre méthode de fusion de régions est composée de trois étapes. À chaque étape, des régions sont fusionnées itérativement de petites à grandes

jusqu'à convergence. Des scores de fusion sont calculés et seuillés à chaque étape. Des conditions d'arrêt sont utilisées pour empêcher le sur-fusionnement. Les scores de fusionnement sont calculés à partir des caractéristiques suivantes des régions: similitude des niveaux de gris, taille de région et longueur de la frontière commune. La première étape favorise la fusion de régions similaires et de longue frontière commune. La deuxième étape favorise des régions de grande différence de taille ainsi que de longue frontière commune. La troisième étape est une relaxation de l'étape deux pour former des régions encore plus grandes - favorise des régions de grande différence de taille ou de longue frontière commune.

3 Suivi de régions dans les vidéos

3.1 Modèle de "snake" traditionnel

Les "snakes" traditionnels appartiennent aux contours actifs paramétriques. Le contour est déformé sous l'influence de trois forces dans une formule paramétrique [3]. Des forces internes sont conçues pour tenir la courbe ensemble (des forces d'élasticité) et pour l'empêcher de plier trop (les forces de recourbement). Les forces de l'image sont les caractéristiques dans l'image qui attirent le "snake" (par exemple, les arêtes ou le gradient). Des forces additionnelles sont utilisées parfois en tant que forces externes pour contraindre la déformation du "snake", mais elles ne sont pas utilisées dans ce travail.

3.2 Vecteurs de flux du gradient (Gradient vector flow)

La différence entre le "snake" GVF et le "snake" traditionnel est dans la force de l'image. Le "snake" traditionnel utilise le gradient d'intensité comme force de l'image. Les forces pour GVF sont dérivées d'une opération de diffusion sur la carte des arêtes et tendent à se prolonger très loin du centre des arêtes. Le procédé informatique de diffusion présenté dans [6] transforme le champ d'écoulement des vecteurs de gradient en le maintenant égal au gradient des arêtes aux frontières et en le laissant changer lentement dans les régions homogènes de l'image. Dans une carte de gradient, les vecteurs de force d'image ont la

grande amplitude à la proximité des bords mais s'éteignent rapidement loin du centre des arêtes (presque zéro dans des régions homogènes). Le champ de GVF prolonge la carte de gradient plus loin à partir des arêtes et dans les régions homogènes. Le "snake" traditionnel a une portée très petite de capture et il doit être initialisé près de la frontière de l'objet. Par contraste, le "snake" GVF peut être initialisé loin de la frontière.

3.3 Application des "snakes" pour le suivi de la cavité

D'une manière générale, le "snake" GVF donne de meilleurs résultats que le "snake" traditionnel dans des applications de suivi d'objets, parce qu'il a une portée étendue de capture et peut détecter des concavités de frontière. Dans ce travail, nous avons besoin du "snake" pour faire le suivi de la région de la cavité dans les images endoscopiques. Dans nos expériences, nous avons constaté que la cavité segmentée inclut souvent des parties provenant des structures voisines qui engendrent des concavités. Les concavités seront gardées par le "snake" GVF, ou même de nouvelles seront créées plus tard pendant la déformation du contour. Pour diminuer cet effet non désiré sur le suivi, nous utilisons d'abord un "snake" traditionnel pour réguler le contour et le déplacer près des arêtes, et ensuite nous utilisons le "snake" GVF pour déplacer le contour sur des arêtes plus forte avec une portée plus grande.

4 Résultats et discussions

Des expériences ont été effectuées à l'aide d'une application développée pour ce travail. L'application est implantée en C++, et elle utilise quelques fonctions de traitement d'images d'une bibliothèque "open source" (OpenCV, [70]). Nous avons également intégré l'algorithme de segmentation par graphe (variante efficace de RSST) [5] et la fonction GVF [71] dans l'application. L'entrée de l'application est une image ou une vidéo. Nous présentons principalement les résultats obtenus à partir d'une vidéo endoscopique chirurgicale (format AVI, modérément compressé) fournie par l'Hôpital Sainte-Justine pour valider chaque phase de notre méthode et pour discuter l'effet de

différents choix de paramètres. La vidéo contient 850 images, et chaque image est de 352 x 240 pixels avec 256 couleurs. Les résultats sont évalués et comparés qualitativement basés sur la perception humaine ainsi que sur les avis et segmentations d'un chirurgien.

Les résultats du prétraitement individuel montrent l'efficacité de chaque étape de prétraitement. Après avoir appliqué les quatre étapes de prétraitement séquentiellement sur les images d'essai, les régions significatives de tissus sont mieux séparées et plus uniformes, ce qui est bénéfique pour l'algorithme de segmentation. L'analyse des résultats de la segmentation par graphe et par fusion en plusieurs étapes montre qu'une segmentation par graphe plus fine ne peut pas améliorer la qualité des résultats de puisque les tissus sont sur-segmentés. Cependant, notre méthode par fusion en plusieurs étapes aide à la division de l'image en des structures ou des objets (bloc de tissus, de cavité, et d'instrument) plus significatifs. Comparé à la segmentation par un chirurgien, nos résultats de segmentation n'ont pas toujours la cavité segmentée correctement. En général, certaines parties de la cavité sont manquantes. En fait, il est très difficile d'obtenir la segmentation désirée pour la cavité. Nous considérons que ce que nous obtenons est une bonne approximation, et une première étape importante pour la segmentation exacte de la cavité, car le résultat que nous trouvons correspond à la partie principale de la cavité. Les résultats sur le suivi par "snakes" de la cavité prouvent que le "snake" GVF peut déformer efficacement le contour pour l'adapter à la cavité de l'image courante, alors que le "snake" traditionnel donne de mauvais résultats lorsque appliqué seul. Nous avons obtenu de meilleurs résultats en utilisant une combinaison des deux "snakes", avec un poids relativement plus grand sur la force interne dans le "snake" traditionnel, et un poids relativement plus grand sur la force externe dans le "snake" GVF. Les "snakes" combinés conservent une portée étendue de capture sans souffrir de bruit de propagation.

La complexité du prétraitement est $O(n)$, où n est le nombre de pixels dans une image. La complexité de la segmentation par graphe est $O(n \log n)$. Chaque étape de l'algorithme de fusion de région en plusieurs étapes est semblable à l'algorithme de RSST, et a une complexité de $O(k_1 m^2)$, où m est le nombre de régions dans la carte de

segmentation d'entrée (beaucoup plus petit que n), et k_1 est un facteur. La complexité de l'algorithme de "snake" est $O(k_2 n)$, où k_2 est un facteur lié au critère d'arrêt et à la taille de fenêtre. Le "snake" GVF est plus lent que le "snake" traditionnel, car le calcul du champ GVF est plus complexe que le champ de gradient.

5 Conclusion et travaux futurs

Dans cette thèse, un système pour faire le suivi de la région formée par une cavité dans des vidéos endoscopiques a été présenté. Nous avons divisé nos objectifs en deux sous-tâches: (1) Recherche de la frontière précise de la cavité dans une image, et (2) suivi de la frontière de cavité d'image en image dans la vidéo. La première sous-tâche a été accomplie au moyen de prétraitements et d'une segmentation par graphe suivie d'une segmentation par fusion de régions en plusieurs étapes. La deuxième sous-tâche a été réalisée au moyen d'une combinaison de "snake" traditionnel et de "snake" GVF. Les images endoscopiques obtenues dans le contexte de discectomie thoracique comportent plusieurs défis. Les résultats expérimentaux montrent que notre méthode de fusion par plusieurs critères peut améliorer les résultats de segmentation de manière significative et le processus intégré de segmentation peut segmenter la cavité adéquatement pour une variété d'images difficiles. Ils montrent également que le "snake" traditionnel combiné au "snake" GVF peut efficacement faire le suivi de la cavité dans un certain nombre d'images.

Nous avons également conçu une logique de suivi perdu pour permettre une remise en marche facile de la segmentation et du suivi en cas de coupure dans la scène ou d'autres événements spéciaux. La contribution principale dans ce travail est la segmentation par fusion de régions en plusieurs étapes conçue pour surmonter les inconvénients habituels de la segmentation basée sur les graphes et pour aider à segmenter la région de cavité nécessaire au suivi. Une deuxième contribution importante est la combinaison de deux types de "snakes" pour conserver une portée étendue de capture sans souffrir de bruits de propagation pendant le processus de suivi.

Les développements futurs de ce travail pourraient être l'amélioration de la qualité et la robustesse de la segmentation en intégrant l'information temporelle et en utilisant des prétraitements adaptés, l'amélioration de l'efficacité de la segmentation par fusion de régions en plusieurs étapes en optimisant le code, et l'amélioration de la qualité et la robustesse du suivi cheminement en repensant le design de la fonction GVF. En outre, un suivi entièrement automatique pourrait être mis au point en sélectionnant un critère unique à la cavité.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

RSST:	Recursive Shortest Spanning Tree
GVF:	Gradient Vector Flow
SST:	Shortest Spanning Tree
EM:	Expectation-Maximization
MAP:	Maximum A Posteriori
CT:	Computer Tomography
MRI:	Magnetic Resonance Imaging
PET:	Position Emission Tomography
OpenCV:	Open Computer Vision Library

LIST OF APPENDICES

INTRODUCTION

Endoscopy is a minimally invasive surgical procedure that utilizes multiple small incisions on the patient's body through which the surgeon inserts tools and a video scope for conducting an operation (Figure 0.1). The scope acquires images of internal organs and of the instruments; and the surgeon performs the operation by viewing the scope images displayed on a video monitor. The advantages of this kind of procedure are minimal inconvenience to patients, and improved diagnostic accuracy and therapeutic outcome.

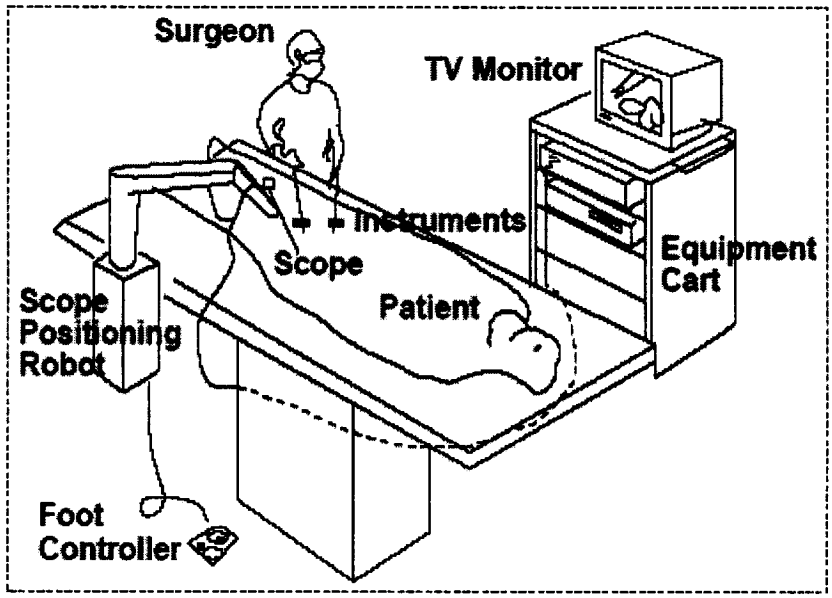


Figure 0. 1 Illustration of endoscopic surgery (source: [1])

Research has been conducted in recent years to process endoscopic surgical images for improving the quality of operations. Most researchers focused on segmenting the instruments in the images and tracking their movements. A good segmentation of surgical instruments has potential usage on automatic scope positioning during endoscopic surgery, which will improve the ease of the surgeon to control his visual feedback. Uecker et al. have conducted intensive research on locating and tracking instruments in abdominal endoscopic surgical video sequences [1, 2]. They reported that their algorithm "seems to be adequate for localizing and tracking instruments in

endoscopic images," and "has been subject to rigorous test in the operating room environment."

In our case, we are interested in images from an application with different properties. The images we are dealing with are thoracoscopic images. The thoracic discectomy uses the technique of endoscopy to remove inter-vertebral disc from the spine of patients suffering from scoliosis. This procedure is performed to fuse some vertebral levels in order to stop the progression of spinal deformities. Compared to invasive surgeries, this method reduces patient risk and costly in-hospital recovery periods. Unfortunately, some challenges still need to be overcome to facilitate the adoption of this procedure. By using an endoscope, the surgeon loses the depth perception. Furthermore, context depth cues are scarce. Hence, a lot of training is necessary for surgeons before practicing this procedure on a patient.

To help the surgeon, the goal of this work is to localize the boundary of the removed disc, which is identified in the image as a cavity, to allow its 3D reconstruction. Then the relative distance between the instrument and the spinal cord is estimated based on the 3D registration of the cavity with a preoperative 3D model of the spinal cord obtained from MRI images. In this thesis, the specific objectives are:

- Segmentation of the cavity from the other tissues and instruments in thoracoscopic images;
- Tracking of the cavity to precisely locate the cavity area from frame to frame.

Endoscopic images obtained in the context of thoracic discectomy are very challenging because limited illumination causes rapid intensity fall-off with radial distance from the image center, polished instruments and moist tissues cause strong specular reflections, motion and focusing of the endoscope cause large changes in apparent size of objects, and exposed tissues or cuts cause variations in textures and colours. For sake of simplicity, we generally call the images that we deal with endoscopic images in the rest of the thesis. Our review of related work has shown that no existing methods are suited for our specific application. For example, some researchers are using

statistical learning based methods to locate and track instruments in endoscopic images. Although some obtained promising results, we cannot just simply adopt the same method, because the properties of the concerned objects are different. Instruments are rigid objects and have regular shapes, which is a very good feature for building a model and for learning by a specific learning algorithm. On the contrary, the shape of the cavity is irregular, and it changes from frame to frame depending on the focus of the scope. The variable shape of the cavity naturally leads us to deformable models. Since deformable models are able to represent complex shapes, they have been applied to medical images generated by X-ray, CT (Computer Tomography), MRI (Magnetic Resonance Imaging), angiography, ultrasound and other sources, and they have been used to segment, match and track a broad variability of anatomic structures [4]. In addition, deformable models are computationally cheap compared to many other image processing operations.

A deformable contour is a planar curve that has an initial position and an objective function associated with it. A special class of deformable contours called snakes was introduced by Kass et al. [3] in which the objective function is referred to as the energy of the snake. By analogy to physical systems, the snake slithers to minimize its energy over time. In practice, a snake must be initialized close to the structure of interest to guarantee good performance. The user can draw the initial contour manually, but that is tedious and error-prone. Based on the deformable contour application model, we have designed a more efficient framework incorporating segmentation and tracking to fulfill our objective. The typical scenario of our application is described as follows. First a video sequence is loaded and a starting frame is selected, then the system performs a dedicated segmentation on that frame that should clearly distinguish the cavity from other tissues. After that, the cavity area is simply specified by a click in the segmentation map, and then the system will go over the rest of the frames to track the specified area by means of a snake method.

In this thesis, we present a dedicated segmentation algorithm that integrates graph-based segmentation and multistage region merging, as well as an application of deformable models – tracking a contour using a combination of traditional snake and

gradient vector flow (GVF) snake [6]. Our particular contribution is a multistage region merging process that follows a coarse graph-based segmentation to overcome usual graph-based segmentation drawbacks.

The outline of our approach is as follows. As first step of the process, frames are extracted from the video sequence and pre-processed. The pre-processing includes Gaussian smoothing, brightness and contrast enhancement, colour space conversion and specular reflections removal. Then a graph-based segmentation is applied on the pre-processed image to obtain a relatively coarse segmentation. The segmentation method is based on a recently proposed approach [5] that is an efficient variant of the recursive shortest spanning tree (RSST) algorithm. After that, the segmentation map is treated by a multistage region merging algorithm. The graph-based segmentation algorithm uses similarity on grey-level as the criterion to group pixels into homogeneous regions, whereas the region merging algorithm uses different criteria at different stages to merge regions into more meaningful structures or objects (e.g., block of tissues, cavity, and instrument). The criteria are selected by making use of prior knowledge or reasonable hypotheses that will be described later. We use a formula to combine different criteria for computing a merging score, then decide whether to merge or not by a threshold.

The contour tracking of the specified area is performed by means of a snake model. First, the same pre-processing is applied on current frame, and the gradient vector flow is computed from the pre-processed image. The contour from previous frame or specified by the user is then evolved to the new boundary in current frame through the energy minimizing process of traditional snake. Next, the new boundary is deformed again in the guidance of GVF snake. The result of GVF snake is interpolated to form a dense contour, which is used as the initial contour of next frame. Therefore the original contour is tracked from frame to frame as the process continues. The snakes we used belong to parametric active contours. The contour is deformed under the influence of three forces in a parametric formula. Internal forces are designed to hold the curve together (elasticity forces) and to keep it from bending too much (bending forces). Image forces are features in the image that attract the snake (e.g., edges or gradient). Additional

forces sometimes are used to constrain the deformation of the snake. The difference between GVF snake and traditional snake is in the image force. The traditional snake uses intensity gradient as the image force. The GVF forces are derived from a diffusion operation and tend to extend very far away from the object. GVF forces can also pull active contours into concave regions. That is often mentioned as strength of GVF snake, but it is actually a disadvantage for our application, because we do not expect the cavity to extrude into concave regions of neighbouring structures. So we first use a traditional snake to regulate the contour and move it to the edges in close vicinity, then use GVF snake to further move the contour to strong edges in a larger range.

The thesis is organized as follows. In Chapter 1, we present an overview of classical and recently proposed techniques for image segmentation and object tracking. We also present a brief survey of image processing applications on medical images, and explain our research objective in detail. The background (theory) of adopted graph-based segmentation algorithm and snakes tracking algorithm is given at the end of this chapter. Chapter 2 is dedicated to the methodology used in this work. In particular, we will describe the consideration on feature selection, discuss the image pre-processing methods specially tailored for the endoscopic images, present the application of the adopted graph-based segmentation algorithm and demonstrate the rationale of our novel multistage region merging algorithm. We also present the solution proposed for the achievement of temporal tracking of specified region among adjacent frames. We describe the application of the snakes in our tracking system, as well as some special processes such as interpolation of deformed contour and detection of lost tracking. In Chapter 3, we present our experimental results of the segmentation algorithm and tracking algorithm on endoscopic images, and discuss the effect of different parameters setup. Finally, Chapter 4 ends this thesis with concluding remarks and a brief discussion of possible avenues for future research.

Chapter 1 State of The Art

In this chapter, we provide an overview of classical and recently proposed techniques for image segmentation and object tracking. Among those techniques, some are adopted and augmented in our research and their backgrounds are described in detail. This chapter is organized as follows: we will review the most typical approaches to image segmentation and moving object tracking in the first two sections. Next, in section 1.3 we present a brief survey of image processing applications on medical images, and discuss several applications on endoscopic images. We then explain our research objective and some considerations about method selection in section 1.4. Finally, we describe the backgrounds of adopted segmentation and tracking techniques in the last section. It should be noted that in video, tracking and segmentation are intimately linked as results from segmentation can be used for tracking, and detected motion by tracking can be used for segmentation.

1.1 Image segmentation methods

The purpose of image segmentation is to decompose the image into non-overlapping regions that are meaningful with respect to a particular application. Numerous segmentation methods have been proposed in the literature, we summarize seven basic methods as follows. Interested readers are referred to Appendix I for more detailed introduction about these segmentation methods.

1) Histogram thresholding.

Thresholding is perhaps the most intuitive form of image segmentation. An image can be simply segmented by thresholds of pixel intensity value. In histogram-based thresholding, the image histogram is used for setting various thresholds to partition the given image into distinct regions. Thresholding is simple, computationally cheap and fast. That is why it is the oldest segmentation method and it is still widely used in simple applications. In complex applications such as medical image processing, finding the appropriate thresholds is often very hard or even impossible. Besides, another drawback of thresholding based segmentation methods is that they do not consider the spatial

information, i.e. the adjacency of pixels. Image areas with similar characteristics are grouped together independently of their location in the image, and this could easily give a result of a segmented object composing of many different regions scattered in the image. So basically segmentation based on thresholding could give correct results only in quite particular cases. In this work, we used histogram thresholding to remove some undesired high intensity pixels during image pre-processing; it is quite simple and efficient. However, we cannot rely on this method for the segmentation of the cavity for the aforementioned reasons.

2) Edge-based segmentation.

Edges are pixels where an image function (e.g. brightness) changes abruptly. A change of the image function can be described by a gradient that points in the direction of the largest growth. The gradient is approximated in digital images by a convolution with edge detection operators [8, 16]. Edge-based segmentations rely on edges found in an image by edge detection operators – these edges mark image locations of discontinuities in gray level, color, texture, etc. Image resulting from edge detection cannot be used as a segmentation result. Supplementary processing steps must follow to combine edges into edge chains that correspond better with borders in the image. A typical edge-based segmentation process is found in [15], in which a set of algorithms were proposed to perform segmentation of natural scenes through boundary analysis.

Although edge detection is well studied and some edge detectors are efficiently used in practice, edge-based segmentation is not an easy task. Edge properties have to be analyzed to assemble local edges into meaningful boundaries that define distinct regions. The more prior information that is available to the segmentation process, the better the segmentation results that can be obtained [8]. In this work, we did not use edge-based method as our segmentation method, but we used canny edge detector throughout our application to extract boundaries in the segmentation map. We rejected this method, because we are more interested in finding regions than finding boundaries. Although regions and boundaries are related, the criteria used to construct them are different. Since cavity regions are not uniform, a focus on boundary will give rise to intra-cavity edges. It

is easier and more natural to express our problem in terms of regions and to use criteria on regions.

3) Region-based segmentation.

Region based segmentation methods are generally better in noisy images where edges are difficult to detect, because noisy pixels giving rise to spurious edges can be integrated into regions following the properties of the neighbouring pixels. Region growing algorithms are based on the growth of homogeneous regions according to certain features (properties) as intensity, colour or texture. Region splitting is the opposite of region merging. It begins with the whole image represented as a single region. The existing image regions are sequentially split to satisfy the given criteria of homogeneity. A combination of splitting and merging may result in a method with the advantages of both approaches. Graph theory is frequently referenced in region-based segmentation methods. It regards each pixel as a node in a graph. An arc joins neighbouring pixels whose properties are similar enough. An efficient graph-based segmentation method is adopted in this work, and the details are given in section 1.4.

Since we want to use application-based knowledge in the segmentation process, split and split and merge method, which are typically based on low level information, do not have any advantages over merge method in this context. Indeed, for splitting, region properties are composite properties of grouped regions and they are not explicitly available as for merging methods. In fact, application-based knowledge is easier to integrate for merging than for splitting. Brox et al. [66] applied a multistage merging process based on Ward, Mean-Ward and border criteria following a watershed pre-segmentation. Xuan et al. [67] used a merge score based on grey-level similarity, region size and region connectivity in their MR brain image segmentation. Van Droogenbroeck et al. [68] explored texture features for their merging criteria. We also designed a multistage merging algorithm incorporating application specific knowledge to process the endoscopic images; details are given in Chapter 2.

4) Cluster-based segmentation.

One natural view for segmentation is to determine which components of a data set naturally "belong together". This is a problem known as clustering. There are two basic algorithms for clustering. In divisive clustering, the entire data set is regarded as a cluster and clusters are recursively split to yield a good clustering. In agglomerative clustering, each data item is regarded as a cluster and clusters are recursively merged to yield a good clustering.

Three commonly used clustering algorithms are the K-means or ISODATA algorithm [45], the fuzzy c-means algorithm [34], and the expectation-maximization (EM) algorithm [46]. The K-means clustering algorithm clusters data by iteratively computing a mean intensity for each class and segmenting the image by classifying each pixel in the class with the closest mean. The fuzzy c-means algorithm generalizes the K-means algorithm, allowing for soft segmentations based on fuzzy set theory. The EM algorithm applies the same clustering principles with the underlying assumption that the data follows a Gaussian mixture model. It iterates between computing the posterior probabilities and computing maximum likelihood estimates of the means, covariances, and mixing coefficients of the mixture model.

Obviously, the idea of clustering is quite similar to region-based segmentation methods. The difference between the two classes of segmentation methods is twofold. First, clustering is considered in a feature space in which multiple image features are extracted from small local areas of the image, but region merging or splitting is often performed with one global feature such as grey level. Second, clustering algorithms do not take into account the spatial adjacency of pixels. In other words, partition or grouping pixels in a cluster is not based on the similarity of adjacent components as in region-based methods. Hence clustering methods have the same drawbacks as histogram based segmentation method.

5) Segmentation by mathematical morphology.

Mathematical morphology uses set operations to extract image components. Most morphological operations are based on simple expanding (dilation) and shrinking (erosion) operations. The basic morphological tool for image segmentation is watershed

[24]. Morphological segmentation actually relies on two basic steps: the first, known as marker extraction aims at identifying a set of initial homogeneous zones (whose union does not cover the entire image, but leaves areas of uncertain attribution). The second decision step is based on the watershed process, which attributes the remaining pixels to the appropriate zones. The name of the technique refers to the analogy that can be drawn between the scheme of the algorithm and the process of filling with water a number of adjacent valleys and thus identifying the edges that separate them. Segmentation algorithms based on watershed are reported among others in [25, 26]. A further extension of morphological watershed is presented in [27], where the method is applied in a multidimensional feature space.

The idea of watershed segmentation is somewhat similar to the region growing method. We would have considered it as an alternative of our graph-based segmentation algorithm if it had the same efficiency. In this work, we used morphological operators in multistage region merging algorithm to find the adjacent regions of one specific region.

6) Segmentation by statistical methods.

Statistical segmentation methods utilize the techniques developed in the field of statistical pattern recognition to classify pixels into different regions. The recognition system works in two modes. In the training mode, the feature extraction module finds the appropriate features for representing the input patterns and the classifier is trained to partition the feature space. In the classification mode, the trained classifier assigns the input pattern to one of the pattern classes based on the measured features.

Statistical methods are widely used in many fields. We also see some applications in image segmentation [22, 29, 30], especially in endoscopic image analysis [1, 32, 33]. A disadvantage of statistical methods is that they are basically supervised methods since they require training data that are often manually segmented and then used as references for automatically segmenting new data. In addition, like clustering methods, they generally do not perform any spatial modeling.

7) Segmentation by detecting an object with template matching.

Sometimes the purpose of segmentation is to detect a specific object in the image. If a template describing a specific object is available, object detection becomes a process of matching features between the template and the image sequence under analysis. Object detection with an exact match is generally computationally expensive and the quality of matching depends on the details and the degree of precision provided by the object template. Fixed templates are useful when object shapes do not change in time and with respect to the viewing angle of the camera.

Two major techniques have been used in template matching. In block matching, the template position is determined from minimizing the distance function between the template and various positions in the image. This method is simple and performs well only in restricted environments where imaging conditions, such as image intensity and viewing angles between the template and the processed images are the same. An application of this method is described in [38]. Matching by correlation is a more complex template matching method. It utilizes the position of the normalized cross-correlation peak between a template and an image to locate the best match. This technique is generally immune to noise and illumination effects in the images, but suffers from high computational complexity caused by summations over the entire template. Point correlation can reduce the computational complexity to a small set of carefully chosen points for the summations [39].

One of our objectives is to detect the cavity area in the endoscopic image, but unfortunately, in our application, a fixed template corresponding to the cavity cannot be formulated as its shapes vary greatly, and its colour and textures vary depending on cut and blood spreading.

1.2 Moving object tracking techniques

Moving objects are always associated with videos as a form of storage. Videos are actually sequences of images, each of which called a frame, displayed in fast enough frequency so that human eyes can perceive the continuity of its content. It is obvious that all image processing techniques can be applied to individual frames. Because in a video sequence images are taken at closely spaced time intervals, the contents of two

consecutive frames are usually closely related. So in addition to all image features used in (still) image segmentation, there should be more information available by analyzing the temporal correspondence between frames. This temporal feature in video sequences is usually called motion information. Note that some methods rely only on motion information for segmentation, but the best performing methods use still image segmentation followed by a temporal validation. Here, we will discuss both segmentation based on motion information, and tracking based on temporal information. Since we are interested in segmenting out a cavity coming from a video recorded with an endoscope, we need to investigate how temporal information can help this segmentation.

1) Segmentation using motion information.

By computing motion information, it is possible to differentiate the moving parts in an image from the background, especially when salient moving data is captured. The basis of temporal based segmentation is the motion information. Two models are most widely adopted for the description of motion information, namely global motion model and optical flow model. Interested readers are referred to Appendix II for more detailed introduction about the description of motion information.

Global motion analysis is concerned with modeling the motion that applies to the whole or the majority of the frame. The motion of small regions within a frame is often called object motion while global motion generally describes the motion of the camera. For example the camera may be zooming on a street scene while vehicles move independently within it. The camera may therefore be considered as a dynamic observer moving unrestricted in 3D space. Many modes of motion need to be considered to quantify camera movement: pan, parallel translation, tilt, zoom and rotation. Modeling camera motion is concerned with describing the projection of this movement onto a 2D plane. Global motion can also model movement not related to the camera or object motion, but motion related to the frame, e.g. shear.

Optical flow is an approximation of the local image motion based upon local derivatives in a given sequence of images. That is, in 2D it specifies how much each image pixel moves between adjacent images while in 3D it specifies how much each

volume voxel moves between adjacent volumes. The 2D image sequences generally are formed under perspective projection via the relative motion of a camera and scene objects. The 3D volume sequences were formed under orthographic projection for a stationary sensor and a moving/deforming object. In both cases, the moving patterns cause temporal varieties of the image brightness. It is assumed that all temporal intensity changes are due to motion only.

Generally we can detect the moving parts using any kind of segmentation techniques applied on the motion information, but it is usually not enough to segment the image only by motion data since motion information is known to be inaccurate in uniform areas (imagine the movement of a chameleon taking the colour of its surroundings, it is hard to detect). Besides, motion data tends to be noisy on edges since gradient-based estimation methods typically impose smoothness constraints on edges, which results in an estimation error when different motion parts overlap [34]. On the other hand, segmentation on spatial information (e.g., grey level and colour) is accurate in uniform areas and on edges, but it tends to yield over-segmentation in textured areas. So a fusion of motion information and spatial information, or spatio-temporal information, is adopted by many researchers in segmentation of video sequences [22, 29, 34, 35]. Although spatio-temporal segmentation appears to have good results with some sequences, it is not necessary to work well with endoscopic images. In particular, we found that the motion information we obtained is quite unreliable mainly because the endoscopic images are relatively uniform. We will further discuss it and give examples in Chapter 2.

2) Tracking by deformable models.

Deformable models are suited for cases where objects vary due to rigid and non-rigid deformations. These variations can be caused by either the deformation of the object itself or just by different object pose relative to the camera. Because of the deformable nature of the cavity in the endoscopic image, deformable models are most appealing to tracking it. Deformable models like snakes allow implicitly using motion information by deforming the object boundary found in the previous frame to fit in current frame. Since

from frame to frame motion is usually small, the change between neighbouring fames is small and hence the deformable model can track the boundary of a region corresponding to a segmented object while accounting for change in time and motion. In this approach, a contour of the object under consideration is originally identified, for example by a still image segmentation algorithm. A transformation on the prototype contour is applied to deform it to fit salient edges in the input image. An objective function with transformation parameters that alter the shape of the contour is formulated reflecting the cost of such transformations. The objective function is minimized by iteratively updating the transformation parameters to best match the object [40].

Segmentation using motion information generally uses region-based segmentation techniques to locate/detect moving objects, while deformable models rely on the information provided by the object boundaries. Deformable models have been widely adopted in object tracking because the boundary-based features (edges) provide reliable information that does not depend on the motion type, or object shape. Usually, the boundary-based tracking algorithms employ active contour models, like snakes and geodesic active contours. These models are energy-based or geometric-based minimization approaches that evolve an initial curve under the influence of external potentials, while it is being constrained by internal energies.

Snakes are parametric active contours models used for boundary tracking which was originally introduced by Kass et al. [3]. Snakes move under the influence of image-intensity "forces," subject to certain internal deformation constraints. In segmentation and boundary tracking problems, these forces relate to the gradient of image intensity and the positions of image features. One advantage of the force-driven snake model is that it can easily incorporate the dynamics derived from time-varying images. The snakes are usually parameterized and the solution space is constrained to have a predefined shape. So these methods require an accurate initialization step since the initial contour converges iteratively toward the solution of a partial differential equation. Considerable work has been done to overcome the problems associated with the solution of the equations of motion and to improve robustness to image clutter and occlusions. Curwen and Black

[41] proposed a B-spline representation of active contours, Dubuisson et al. [42] employed polygonal representation in vehicle tracking problems, and Xu and Prince [6] introduced a new external force field known as GVF field to give the snake a large capture range and make it capable of moving into boundary concavities. In this work, we used a combination of traditional snake and GVF snake to track the cavity area. Details are given in Chapter 2.

Deformable models start with a closed curve in two dimensions (or a surface in three dimensions) and allow the curve to move perpendicular to itself at a prescribed speed. One way of describing this curve is by using an explicit parametric form, which is the approach used in snakes. The implicit active contour, or level set approach [43], instead of explicitly following the moving interface itself, takes the original interface and embeds it in higher dimensional scalar function, defined over the entire image. The interface is now represented implicitly as the zero-th level set (or contour) of this scalar function. The snake model is not only compact, but is robust to both image noise and boundary gaps as it constrains the extracted boundaries to be smooth. However, it can severely restrict the degree of topological adaptability of the model, especially if the deformation involves splitting or merging of parts. In contrast, level sets model is designed to handle topological changes naturally. However, unlike the parametric form, they are not robust to boundary gaps and suffer from several other deficiencies as well [44]. The cavity area we want to track does not involve splitting or merging of parts, so the snake model is suited for our application.

1.3 Applications on medical imaging

With medical imaging playing an increasingly prominent role in the diagnosis and treatment of disease, segmentation techniques have been applied for extracting clinically useful information about anatomic structures through modalities such as X-ray, CT, MRI, PET (Position Emission Tomography), ultrasound and other modalities.

It is reported in a survey [47] that at least the following techniques have been seen in medical image processing: (1) thresholding approaches, (2) region growing approaches, (3) classifiers, (4) clustering approaches, (5) statistical models, (6) artificial

neural networks, (7) deformable models, and (8) atlas-guided approaches. Most of these techniques belong to the categories presented in section 1.1 and section 1.2. One common observation about the medical application of segmentation methods is that one method used alone can seldom fulfill the task. It is often necessary to use several segmentation techniques from different categories to obtain a useful result. Thresholding is often used in pre-processing, and region growing or watershed method is often used in initial segmentation. Statistical method requires training data and a training process, while clustering method and deformable model needs initialization (or equivalently, initial segmentation). In fact, an integration of different techniques can practically keep or enhance the advantages of some techniques, and eliminate or alleviate the impact of disadvantages of others. This will result in a better output of segmentation at the end. In this work, we used histogram thresholding in pre-processing phase, graph-based segmentation and multistage region merging in segmentation phase, and two types of snakes in tracking phase.

Deformable models have attracted special attention in medical imaging. Two-dimensional and three-dimensional deformable models have been used to segment, visualize, track and quantify a variety of anatomic structures ranging in scale from the macroscopic to the microscopic. These include the brain, heart, face, cerebral, coronary and retinal arteries, kidney, lungs, stomach, liver, skull, vertebra, objects such as brain tumours, a fetus, and even cellular structures such as neurons and chromosomes. Deformable models have been used to track the non-rigid motion of the heart, the growing tip of a neurite and the motion of erythrocytes. They have been used to locate structures in the brain, and to register images of the retina, vertebra and neuronal tissue. A general review on deformable models in medical image analysis can be found in [4].

Although there are numerous applications on medical imaging, few applications on endoscopic surgical images were reported in the literature. Uecker et al. [1] used a Bayesian classifier to segment images into two classes, which are organ and instrument. Lo et al. [32] used a similar method in their tissue/instrument segmentation. Recently, Boisvert et al. [33] used a support vector machine for the classification. Basically those

researchers focused on segmenting the instruments in the images and tracking their movements. The main techniques used in those work belong to statistical methods. There also exist other methods for processing specific endoscopic images. Asari [48] reported a fast and accurate segmentation technique for the extraction of gastrointestinal lumen from endoscopic images. He used a differential region growing technique on the basis of a similarity criterion, and a dynamic hill-clustering method to ensure the effectiveness of the terminating condition during the growth process. Bockholt et al. [49] used statistical texture analysis for bladder tumour detection. Karkanis et al. [50] presented an approach to the detection of tumours in colonoscopic video. It was based on a new colour feature called colour wavelet covariance, which is based on the covariances of second-order textural measures.

1.4 Research objective and considerations about method selection

As mentioned in the introduction part of this thesis, our research objectives are to precisely locate and track the cavity area in the thoracoscopic video sequences in preparation of the 3D reconstruction of this area. To fulfill the objectives, we divided our work into two subtasks: (1) finding the precise boundary of the cavity in a frame, and (2) tracking the cavity boundary from frame to frame. The first subtask was accomplished by means of image pre-processing, graph-based segmentation followed by multistage region merging. The second subtask was accomplished by use of a combination of traditional snake and GVF snake.

We have reviewed image segmentation methods in section 1.1, moving object detecting and tracking techniques in section 1.2, as well as their applications on medical imaging in section 1.3. It seems that there are a lot of methods to choose from; some may be applicable to our application, because they were successfully used in other applications. Undoubtedly there is more than one path to reach our goal. Here we explain our choices.

Why not use statistical methods since three other papers dealing with endoscopic surgical images had all proposed this type of methods? Although they dealt with different types of endoscopic images, they all focused on segmenting the instruments in the images

and tracking their movements. So basically they only need to classify the image into two classes: instruments and tissues. Since instruments have good shape features for learning and for classifying, adopting in this case statistical methods is not a bad choice. Nevertheless, our goal is to locate the cavities that change drastically from frame to frame. It is hard to find a proper feature of the cavities for statistical models. Besides, we also want the segmentation free of learning and to segment a variety of images on the fly, which is difficult for statistical methods.

Why not use motion information? Motion information is indeed an important clue in video sequences. Many video segmentation approaches used spatiotemporal segmentation to locate and track moving objects in video sequences. Unfortunately, the motion data we obtained through different algorithms are generally unreliable mainly because the images are lack of textures. We will show some examples in the feature selection section of next chapter. Instead, we chose to use snakes to account for motion between frames without explicitly computing motion data.

Why use a multistage region merging process, which is not often seen in the literature? In fact, our multistage region merging algorithm is specially designed to overcome the drawbacks of usual region-based segmentation methods. It incorporates some application-based knowledge and uses different merging criteria in three stages to further process the segmentation map in order to identify more meaningful structures or objects, e.g., the entire cavity area that we concerned with. We will talk more about the algorithm in next chapter.

1.5 Backgrounds of adopted segmentation and tracking algorithms

In this work, we adopt the efficient graph-based segmentation algorithm [5] to do an initial segmentation, and use a combination of traditional snake [3] and GVF snake [6] to do region tracking.

1.5.1 Why using graph-based segmentation?

Our segmentation goal is a precise multi-class segmentation of endoscopic images, and we also want the segmentation free of learning and to segment a variety of images on the

fly. Now let us think about what segmentation methods could fit our objective, by briefly reviewing our conclusions from section 1.2.

First of all the simple thresholding based methods are not good enough for a complex scheme since they do not use spatial information. More complex methods like classification and statistical ones cannot fully meet our requirements as mentioned before, although they are versatile and frequently appeared in the literature. Compared to classification and statistical segmentation methods, region based methods are more straightforward in principle and are easier to use in practice. Watershed based segmentation methods are comparable to segmentation based methods, but they are suffering from higher computational cost. As for edge based methods, special processes are needed after the edge detection. Roubert used edge based approach in segmentation of endoscopic images [56], and he also aimed at segmenting instruments as some other researchers did. Because he focused on distinguishing instruments from the background, he used thresholding and window filtering to filter out the regions that were considered non-instruments in the first step, and then he used edge detection and line fitting to label the instruments in the remaining image. His method cannot fit our goal since we cannot reject any region to facilitate line fitting. In addition, line fitting does not apply to segmentation of tissues which are unstructured.

From these drawbacks of others methods, we concluded that region-based methods are more appropriate for our goal. Furthermore, we chose a specific region based segmentation method which is a graph-based approach. It is the most widely adopted and it has been used for decades. Graph-based methods map an image onto a graph where nodes are composed of pixels/regions and edges are composed of links between neighbouring nodes. Each node has a weight based on some feature and each edge has a weight generally defined by the weight difference of the nodes it connects. The algorithm will group nodes [18] or cut the graph into connected regions [60] by edge weight (reflecting similarity of pairs of nodes); it can be used without any supervision, and do not require a learning phase. Graph-based segmentation takes into account global image properties as well as local spatial relationships, and results in a region map that is

ready for further processing, e.g. region merging or labelling. Among many variations of graph-based segmentation methods, RSST based algorithms have been used from 1980s [58]. The algorithm recursively finds the shortest link weight edge and eliminates it (thus the two nodes connected by the edge are grouped in one region). The RSST algorithm groups similar pixels into homogeneous regions in a desired fashion for our application. It has been successfully applied in many segmentation tasks [18, 61, 62]; most recent applications include object-oriented segmentation for next generation video standard [63, 64]. In this work, we adopted a recently proposed efficient graph-based segmentation algorithm [5]. In the following subsections, we first briefly introduce the graph theory and the conventional RSST based algorithm, then we present the efficient graph-based segmentation algorithm.

1.5.2 Graph theory and RSST based algorithm

Graph theory is the study of graphs and their applications. A graph $G = (V, E)$ is made up of a set of vertices v_i and v_j connected to each other by links/edges e_{ij}, for $i \neq j$, where v_i and v_j are the vertices that the link connects. In a weighted graph the vertices and links have weights associated with them, namely, w_i and w_{ij} respectively (also known as vertex weight and link weight, respectively). Now let us see some definitions useful for understanding RSST. Each vertex is not necessarily linked to every other, but if there is a link between every pair of vertices then the graph is complete. A *partial graph* has the same number of vertices but only a subset of the links of the complete graph. A *chain* is a list of successive vertices in which each vertex is connected to the next by a link in the graph. A *cycle* is a chain whose end links meet at the same vertex. A *tree* is a connected set of chains such that there are no cycles. A *spanning tree* is a tree, which is also a partial graph. The *shortest spanning tree* of a weighted graph is a spanning tree such that the sum of its link weights, or some other monotonic function of its link weights, is a minimum for any possible spanning tree. It is trivial if it consists of a single vertex, and it is nontrivial, otherwise. A *forest* is a set of trees, and a *spanning forest* is a forest that is also a partial graph. A full explanation of these terms can be found in [57].

In RSST, a 2D image is described as an undirected graph $G = (V, E)$, in which pixels $v_i \in V$ are mapped to nodes in the graph, and link with neighbouring pixels form the edges $e_k = (v_i, v_j) \in E$. The graph can be 4-connected or 8-connected. It means that a pixel can be only linked to its 4-neighbors or 8-neighbors to form edges. Figure 1.1 illustrates the mapping of a colour image onto a 4-connected graph. A segmentation S is a partition of V into regions such that each region R corresponds to a connected subset of the edges in E. It is conducted by recursively finding the least weight edge and merging the two regions (which are originally pixels) connected by the edge. Figure 1.2 illustrates the process of iterative region merging. It is explained further below.

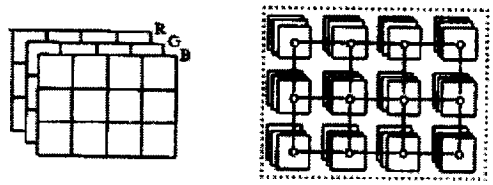


Figure 1. 1 Mapping a colour image (left) onto a graph (right) (source: [58])

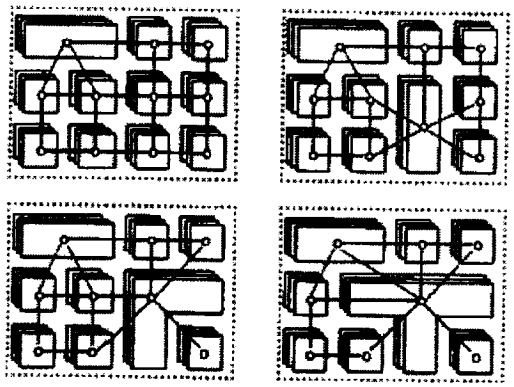


Figure 1. 2 Image segmentation by iterative region merging (source: [58])

The conventional RSST algorithm [58] consists of two functional blocks, namely, the initialization stage and the linking process. The flowchart of the RSST algorithm is depicted in Figure 1.3.

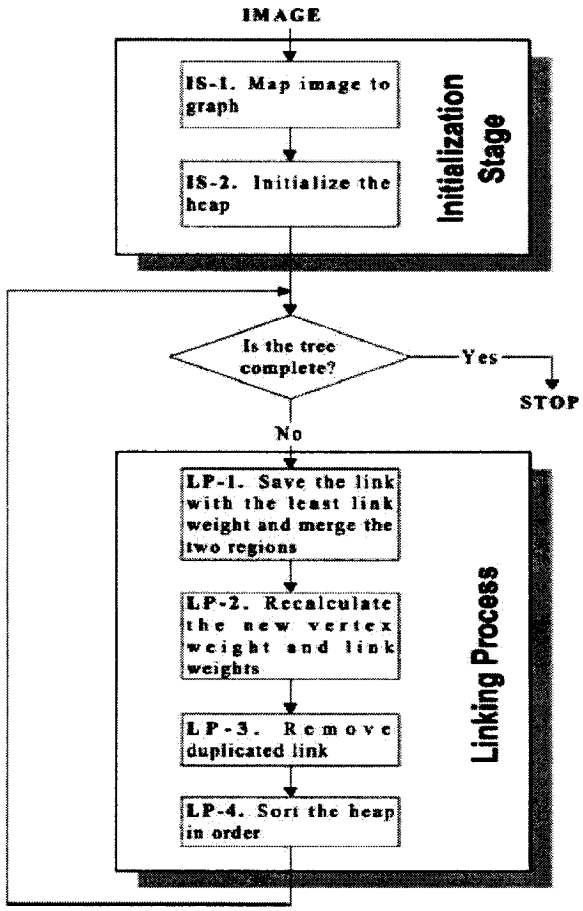


Figure 1. 3 Flowchart of the conventional RSST algorithm (source: [59])

RSST recursively builds the shortest spanning tree (SST). The RSST starts with a mapping of an image onto a weighted graph at the initialization stage of IS-1. Each region or vertex initially contains only one pixel. The pixel intensity values of regions are used to evaluate vertex weights and link weights of the graph. A vertex weight is defined as the average intensity value of the corresponding region, while a link weight is evaluated by a link weight function or a cost function, which is basically a function of the vertex weights and the sizes of the connected regions. All links are then sorted in order according to their link weights, and stored in a heap at IS-2. The linking process is an iterative process to form the SST, so the algorithm got its name RSST. In the beginning of the cycle, a link with the smallest link weight in the graph is chosen to add in the SST. The chosen link is saved and the two connecting or merging regions are merged in LP-1.

The vertex weight of the newly merged region is updated, hence, all surrounding links need to be recalculated in LP-2 and all loop-forming links, also known as duplicated links, will be removed in LP-3. Subsequently, all remaining links are sorted in LP-4. Thus, the number of regions is progressively reduced from a pixel-by-pixel image, down to just one if desired. Those saved links form a spanning tree representation of the image. By noting the order in which the links are saved, a hierarchical representation of the original image is created.

1.5.3 Efficient graph-based segmentation algorithm

Recently, Felzenszwalb and Huttenlocher proposed an efficient graph-based segmentation algorithm [5], which is an efficient variant of the RSST algorithm. The principle of the algorithm is described in the following.

Given a set V of elements (e.g., image pixels) to be segmented, the goal is to find a partition, or segmentation $S = \{C_1, C_2, \ldots, C_p\}$. Denote $D(C_i, C_j)$ a pairwise region comparison Boolean function that evaluates whether or not there is evidence of a boundary between the two components C_i and C_j. Based on the goodness criteria terms "too-fine", i.e. over-segmentation and "too-coarse", i.e. under-segmentation have been formally defined in [5]. S is said to be *too fine* when there is some pair of regions C_1 and C_2 for which $D(C_1, C_2)$ is false. Given two segmentations S and T of V, T is said to be a proper refinement of S when $\forall C \in T, \exists C' \in S$ such that $C \subset C'$. S is said to be *too coarse* when there exists a proper refinement of S that is not too fine. A segmentation is *good* if it is neither too fine nor too coarse. There are several different good segmentations for the same image, and the nature of D dictates some of them.

Further, a graph theoretic approach is proposed which is based on the concept of variation inside a partition and variation across partitions. The algorithm finds the partition S for V such that intensity variation inside any two neighbouring partitions C_i and C_j is less than the variation across them. The partition produced by this algorithm is "neither over-segmentation nor under-segmentation". Intuitively, this algorithm clusters the nodes which are joined by light weight edges (and hence combines "similar pixels" in the original image). For creating these clusters some criteria have been defined, and

clusters are merged if they satisfy those criteria. In the following, we define some terms and the criteria which create the basis for the algorithm.

If S is a segmentation of $G(V, E)$, C is some component in S, $MST(C, E)$ is the minimum spanning tree of C with edges in E, then the algorithm uses following notions of "variation" and "merging criteria":

- Internal variation of a component C is the largest weight in the minimum spanning tree of the component.

$$Int(C) = \max_{e \in MST(C,E)} w(e) \qquad \text{(Equation 1.1)}$$

- Variation/difference between two components C_1, C_2 is the minimum weight edge connecting the two components.

$$Dif(C_1, C_2) = \min_{v_i \in C_1, v_j \in C_2} w((v_i, v_j)) \qquad \text{(Equation 1.2)}$$

- Criteria for merging two components C_1, C_2

$$Dif(C_1, C_2) \leq MInt(C_1, C_2) \qquad \text{(Equation 1.3)}$$

Where

$$MInt(C_1, C_2) = \min(Int(C_1) + \frac{K}{\|C_1\|}, Int(C_2) + \frac{K}{\|C_2\|}) \qquad \text{(Equation 1.4)}$$

$\|C\|$ denotes the size of component C and K is some constant.

We now turn to the segmentation algorithm, which is closely related to Kruskal's algorithm for constructing a minimum spanning tree of a graph [65]. It can be implemented to run in $O(n \log n)$ time, where n is the number of vertices in the graph. Given an input image with a corresponding graph $G = (V, E)$, with n vertices and m edges. The algorithm produces a segmentation S as follows:

0. Sort E into $\pi = (e_1, \ldots, e_m)$, by non-decreasing edge weight.

1. Start with a segmentation S^0, where each vertex v_i is in its own component.

2. Repeat step 3 for $q = 1, \ldots, m$.

3. Construct S^q given S^{q-1} as follows. Let v_i and v_j denote the vertices connected by the q-th edge in the ordering, i.e., $e_q = (v_i, v_j)$. If v_i and v_j are in disjoint components of S^{q-1} and $w(e_q)$ is small compared to the internal difference of both

those components, then merge the two components. Otherwise do nothing. More formally, let C_i^{q-1} be the component of S^{q-1} containing v_i and C_j^{q-1} the component containing v_j. If $C_i^{q-1} \neq C_j^{q-1}$ and $w(e_q) \leq MInt(C_i^{q-1}, C_j^{q-1})$ then S^q is obtained from S^{q-1} by merging C_i^{q-1} and C_j^{q-1}. Otherwise $S^q = S^{q-1}$.

4. Return $S = S^m$.

1.5.4 Why using snakes in region tracking?

A straightforward way for region tracking in a video sequence is to segment each image in the sequence and locate the specific region by correspondence (e.g., centroid of the region). The advantage of this approach is that it can track multiple regions at the same time, but it is time-consuming and it always relies on the quality of segmentation result. For our cavity tracking purpose, it is really not necessary to use the straightforward approach, mainly because the multistage region merging process is relatively slow as well as the output of segmentation algorithm is not stable all the time.

Some other approaches utilize the motion information extracted from two or more consecutive frames to identify the moving objects, but motion information is not reliable in our case (see section 2.1). Certainly there exist other approaches, some useful concepts and moving object detecting and tracking techniques have been presented in section 1.2. Among those approaches, deformable models are widely adopted in tracking of special structures in medical images. We also adopted these models and verified that it is a good approach for tracking the cavity area in our endoscopic video sequences.

In this work, the contour tracking of the specified area is performed by means of snake model [3, 6]. The snakes we used belong to parametric active contours. The contour is deformed under the influence of three forces in a parametric formula. Internal forces are designed to hold the curve together (elasticity forces) and to keep it from bending too much (bending forces). Image forces are features in the image that attract the snake (e.g., edges or gradient). Additional forces sometimes are used as external forces to constrain the deformable of the snake. The difference between GVF snake and traditional

snake is in the image force. The traditional snake uses intensity gradient as the image force. The GVF forces are derived from a diffusion operation and tend to extend very far away from the object. GVF forces can also pull active contours into concave regions. That is often mentioned as strength of GVF snake, but it is actually a disadvantage for our application. In our experiment, we found that the segmented cavity often includes parts extruding into concave regions of neighbouring structures, which is not a desired factor. The extruded parts will be kept by GVF snake, or even be developed further during contour deformation because of GVF snake's ability to pull contours into concave regions. So we first use a traditional snake to regulate the contour, then use GVF snake to further move the contour to strong edges in a larger range. The use of both snakes will be explained in more details in section 3.3.

1.5.5 Traditional snake model

Snakes, or active contours, are curves defined within an image domain which can move under the influence of internal forces coming from within the curve itself and external forces computed from the image data. The internal and external forces are defined so that the snake will conform to an object boundary or other desired features within an image. The idea of snakes has been first proposed by Kass, Witkin and Terzopoulos [3] for the segmentation of objects from 2D and 3D images as well as for their visualization.

The snake is made active by always minimizing its energy function. If it is positioned "close enough" to significant contours in the image, that is, close enough so that potential field forces exist in relation to these contours, the snake will move and ultimately reach a local energy minimum corresponding to that image contour. Therefore, the snake relies on external mechanisms to place it close to salient contours. In this work, a dedicated segmentation algorithm (described in Chapter 2) has been designed to segment an input image into several meaningful objects. The initial contour can be simply specified by a click inside the interested object block (e.g., the cavity area) in the segmentation map.

During the application of snake model to the initial contour, its energy function is minimized. One can view the energy minimization problem as a dynamic one where the

deformable model is governed by the laws of elasticity and Lagrangian dynamics. In such a setting the model evolves according to the forces acting on it and stops if equilibrium of all forces is reached. This equilibrium solution is equivalent to a minimum of the energy function. The function to minimize the energy of the model is sometimes called cost or objective function. The framework of snake model comprises three different aspects: the representation of the model, the energy functional and the method of minimizing the cost function. We describe the snake model in this section using notations and mathematical derivations mostly originating from [3].

1.5.5.1 Representation of the model

Mathematically, snake model in 2D can be represented by a curve v as a function of its arc length s,

$$v(s) = (x(s), y(s)), \qquad \text{(Equation 1.5)}$$

with $s \in [0, 1]$. Setting $v(0) = v(1)$ defines a closed curve; otherwise it defines an open curve. In 3D the model can be represented as a surface v which is a function of two parameters, r and s,

$$v(r, s) = (x(r, s), y(r, s)), \qquad \text{(Equation 1.6)}$$

with $s, t \in [0, 1]$. Both representations require an analytic form of the curve or surface v which is usually not available. Instead, Kass et al. [3] suggested using finite differences to obtain a polygonal approximation of the curve or surface. In 2D, such a discrete model can be expressed as an ordered set of n vertices $v_i = (x_i, y_i)$ with $v = (v_1, \ldots, v_n)$. Again, if $v_1 = v_n$ the contour is closed, otherwise the contour is open. This representation uses a fixed number of vertices in order to obtain an approximation of the curve or surface of the model. In the following we only consider the 2D case.

1.5.5.2 Energy functional

Having represented the contour as $v(s)$, the model is defined as a sum of energy terms in the continuous spatial domain. The energy terms can be categorized as follows:

- Internal Energy (E_{int}): Internal energy is a function of the contour $v(s)$ itself and it specifies the tension and smoothness of the curve. It therefore depends on the internal properties of the contour.

- External energy (E_{ext}): It is derived from the image where the contour resides, and it possesses local minima at the edges or the object boundaries.
- Constraint energy (E_{con}): Constraint energy acts on the contour only if some sort of feedback is provided by a user or a higher-level process.

The mathematical model using above energy terms is

$$E_{snake} = E_{int} + E_{ext} + E_{con} \qquad \text{(Equation 1.7)}$$

The energy terms are further explained in the following paragraphs.

Internal Energy (E_{int}): The internal energy of the contour is composed of two terms, elastic energy and bending energy.

- Elastic energy ($E_{elastic}$): The contour is treated as an elastic rubber band giving it an elastic potential energy. This energy consists of first order derivative of the contour and it discourages stretching by introducing tension in the contour.

$$E_{elastic} = \frac{1}{2}\int_{s} \alpha(s) \mid v_s \mid^2 ds \qquad \text{(Equation 1.8)}$$

where $v_s = \frac{dv(s)}{ds}$. Adjusting the weight $\alpha(s)$ allows us to control the elastic energy along different parts of the contour. However for most applications it is assumed to be a constant α throughout the curve. The energy term is minimized and becomes zero when the first derivative of the contour becomes zero everywhere, signifying that the contour of minimum elastic energy has collapsed to a point. Therefore a contour trying to minimize elastic energy deforms by contracting itself (shrinking).

- Bending Energy ($E_{bending}$): The contour is also considered to behave like a thin metal strip. This is another physical property assigned to the snakes. The contour is expected to be a smooth curve or straight line and to avoid sharp corners. Therefore it is considered to possess bending energy, which is given as the sum of squared curvatures.

$$E_{bending} = \frac{1}{2}\int_s \beta(s) \mid v_{ss} \mid^2 ds \qquad \text{(Equation 1.9)}$$

where $v_{ss} = \frac{d^2 v(s)}{ds^2}$. The second order derivative discourages bending. Sharp corners or points of high curvature are characterized as high frequencies and bending energy is more sensitive (very high) for contours having such sharp corners because the second order derivative will be very high for such contours. Setting β(s) to 0 at a point means that we are relaxing the bending force and allowing that point to take the shape of a corner.

Adjusting the weights α(s) and β(s) controls the relative importance of the elastic and bending energy terms and therefore the internal energy of the contours. The first term of the energy functional can now be written as

$$E_{\text{int}} = E_{elastic} + E_{bending} = \int_s \frac{1}{2}(\alpha \mid v_s \mid^2 + \beta \mid v_{ss} \mid^2) ds \qquad \text{(Equation 1.10)}$$

External energy (E_{ext}): External energy is derived from the image that contains the contour. Let us define a continuous function $E_{image}(x, y)$ so that it takes on its smaller values at the features of interest, such as boundaries. External image energy of the whole contour (E_{ext}) is then defined as

$$E_{ext} = \int_s E_{image}(v(s)) ds \qquad \text{(Equation 1.11)}$$

Therefore the key is in defining a suitable E_{image}. Suppose we want the snake to latch to bright structures in the image. Then E_{image} can be simply defined as $E_{image} = -I(x, y)$ where $I(x, y)$ refers to the gray level values of the image. Reducing such an energy function (i.e. making it more negative) will move the snake towards brighter parts of the image. The other ways to define E_{image} which push active contour towards edges are as follows:

$$E_{image}(x, y) = -\mid \nabla \mathrm{I}(x, y) \mid^2 \qquad \text{(Equation 1.12)}$$

$$E_{image}(x,y) = -|\nabla(G_\sigma(x,y) * I(x,y))|^2 \qquad \text{(Equation 1.13)}$$

where $\nabla I(x,y)$ denotes the gradient of the image and $G_\sigma(x,y)$ is a two-dimensional Gaussian function with standard deviation σ. $G_\sigma(x,y)$ is used as a factor of image intensity, and a larger σ will increase the blurring of the edges thereby increasing the capture range of the snakes.

Constraint energy (E_{con}): The constraint energy is determined by the additional constraints set by the user on the contour. The model given by Kass et al. proposed that snakes be used as a power assisted tool for the user to detect objects in an image. In this case the user provides a feedback to the system that is incorporated in the form of constraint energy. One way of doing this is to allow the user to connect a spring between any point x_1 in the spatial domain and a point x_2 on the snake. A snake trying to reduce its constraint energy is pulled towards the point x_1 defined by the user. Constraint energy can also be interpreted as a kind of prior for the system. In our case, constraint energy is not used as we do not want extra user interaction.

1.5.5.3 Numerical solution to the model

With the energy terms defined above we can now formulate the problem as that of finding a curve $v^*(s)$ so that the energy functional E_{snake} attains a minima where

$$E_{snake} = \int_s \frac{1}{2}(\alpha(s)|v_s|^2 + \beta(s)|v_{ss}|^2) + E_{image}(v(s))ds \qquad \text{(Equation 1.14)}$$

Finding a curve for which E_{snake} has a stationary value is a problem of variational calculus and we can apply the fundamental equation known as Euler-Lagrange Differential Equations to the system giving the following equation [3]:

$$\alpha v_{ss} - \beta v_{ssss} - \nabla E_{image} = 0 \qquad \text{(Equation 1.15)}$$

Solution of the above equation gives the final contour minimizing E_{snake}. Equation 1.15 can be also viewed as a force balance so that the energies are associated with the forces:

$F_{\text{int}} + F_{ext} = 0$, where $F_{\text{int}} = \alpha v_{ss} - \beta v_{ssss}$ and $F_{ext} = -\nabla E_{image}$.

F_{int} and F_{ext} denote the internal and external forces, respectively. In this way the deformation process can be explained based on the interaction of the force terms and the deformation stops when the forces balance each other, i.e., on the location of the boundary. The internal forces discourage stretching and bending while the external force pulls the snake towards the desired image edges.

The first step towards solving the problem in image domain is to discretize the contour. In the discrete model, the contour $v(s)$ is represented by a set of control points $v_0, v_1, ..., v_{n-1}$ which are said to be connected by straight lines giving a piecewise linear curve. Each control point has a position, given by (x, y) coordinates in the image, and a contour is entirely specified by number and coordinates of control points. The adjustment of the contour (deformation) is obtained by moving the control points individually. The energy terms are all converted to the discrete form with the derivatives substituted by finite differences and the integrals substituted by summations. The forces acting along the contour are now calculated at each of these control points separately and the entire curve is declared to achieve a minimum when each of these control points attains local minima.

Having represented the contour as a set of discrete points v_i, Kass [3] suggested to use the following finite difference approximations to estimate v_{ss} and v_{ssss}:

$$v_{ss} = \frac{d^2 v(s)}{ds^2} = \frac{|v_i - v_{i-1}|^2}{ds^2} \qquad \text{(Equation 1.16)}$$

$$v_{ssss} = \frac{d^4 v(s)}{ds^4} = \frac{|v_{i-1} - 2v_i + v_{i+1}|^2}{ds^4} \qquad \text{(Equation 1.17)}$$

The first term in Equation 3.11 is usually called elastic force $F_{elastic} = \alpha v_{ss}$, and the second term called bending force $F_{bending} = \beta v_{ssss}$. From Equation 3.12 we get the x and y component of elastic force as

$$\begin{aligned} F_{elastic\_x}(i) &= \alpha(x(i-1) + x(i+1) - 2x(i)) \\ F_{elastic\_y}(i) &= \alpha(y(i-1) + y(i+1) - 2y(i)) \end{aligned} \qquad \text{(Equation 1.18)}$$

From Equation 3.13 we get the x and y component of bending force as

$$\begin{aligned} F_{bending\_x}(i) &= \beta\,(x(i+2) - 4x(i+1) + 6x(i) - 4x(i-1) + x(i-2)) \\ F_{bending\_y}(i) &= \beta(y(i+2) - 4y(i+1) + 6y(i) - 4y(i-1) + y(i-2)) \end{aligned} \qquad \text{(Equation 1.19)}$$

The external force is $F_{ext} = -\nabla E_{image}$. Its x and y component are given by using the central finite difference operation.

$$F_{ext\_x} = E_{image}(x(i)-1, y(i)) - E_{image}(x(i)+1, y(i))$$
$$F_{ext\_y} = E_{image}(x(i), y(i)-1) - E_{image}(x(i), y(i)+1) \quad \text{(Equation 1.20)}$$

A common way to find the solution of the force equation is to make the snake dynamic by treating v as a function of time t as well as s. Hence the contour is now defined as $v(s, t) = (x(s, t), y(s, t))$. Then the partial derivative of v with respect to time t is set equal to the left hand side of Euler equation.

$$\alpha v_{ss}(s,t) - \beta v_{ssss}(s,t) - \nabla E_{image} = v_t(s,t) \quad \text{(Equation 1.21)}$$

When statistical equilibrium for $v(s, t)$ is reached, the term $v_t(s, t)$ vanishes and a solution to Equation 3.17 is obtained. At this point, the forces cancel each other out, causing zero displacement for the points on the contour and we get a stabilized contour. This acts like a stopping condition.

1.5.5.4 Forces acting on the contour

Now let us take a further look at the effect of individual force acting on the contour. Elastic force is generated due to elastic potential energy present in the contour. It is given as $F_{elastic} = \alpha v_{ss}$. The elastic force at any point in the curve is oriented in the direction of center of curvature at each point on the contour. This causes the contour at these points to expand until the concavity is eliminated. The other action of elastic forces is to shrink the curve and it is responsible to collapsing the contour to a single point in the absence of any opposing force. Figure 1.4 below shows the action of elastic force on the curve. The initial contour was allowed to deform freely under the influence of only elastic force. Notice how the concavities are eliminated and the entire curve is shrinking. If we would allow the curve to move freely for few more iteration, it will eventually collapse to a point.

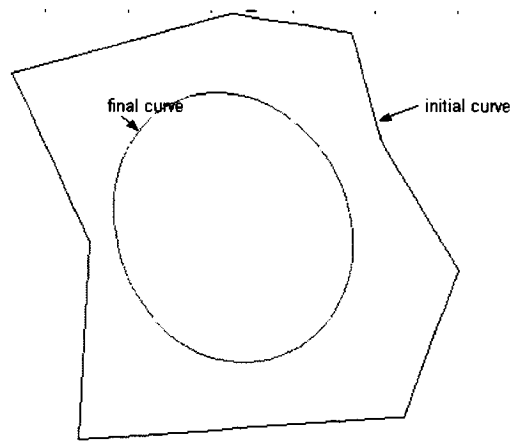
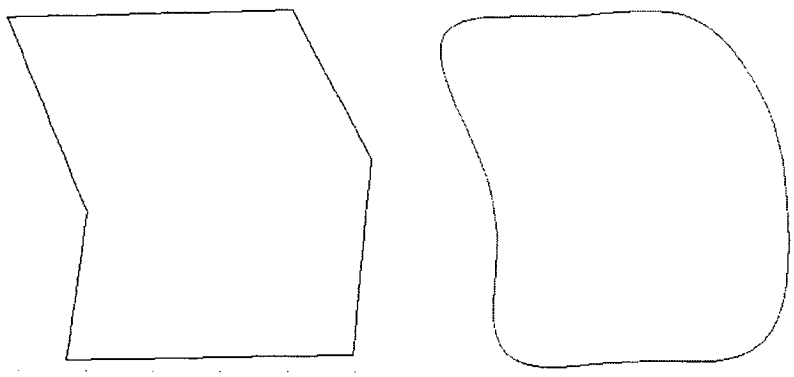


Figure 1. 4 Example of elastic force acting on the contour

The elastic forces also tend to evenly space the control points along the contour making it evenly sampled because a point at equal distance from its neighbours will experience least elastic force due to balancing of the opposite forces from its neighbours. In fact if the three points are collinear and equally spaced the force equals to 0.

Bending force acts on the curve due to the bending energy of the contour and is given by $F_{bending} = -\beta v_{ssss}$. Bending force is strong at corners, i.e., points with very high curvature and it tries to eliminate these sharp corners thereby smoothing the contour.



Initial curve Final curve

Figure 1. 5 Example of bending force acting on the contour

Figure 1.5 helps understand the action of bending force. The initial curve was allowed to deform only under the influence of bending force giving the final curve. Notice how all the corners were smooth out thereby reducing the bending energy of the contour, and this is exactly what bending force should be doing. If we continue for more

iterations the bending force will eventually seek to make the curve into a circle because a circle has lowest bending energy.

The external force is generated due to the potential external energy that we have given to the system. The force seeks to minimize E_{ext} is given as $F_{ext} = -\nabla E_{image}$.

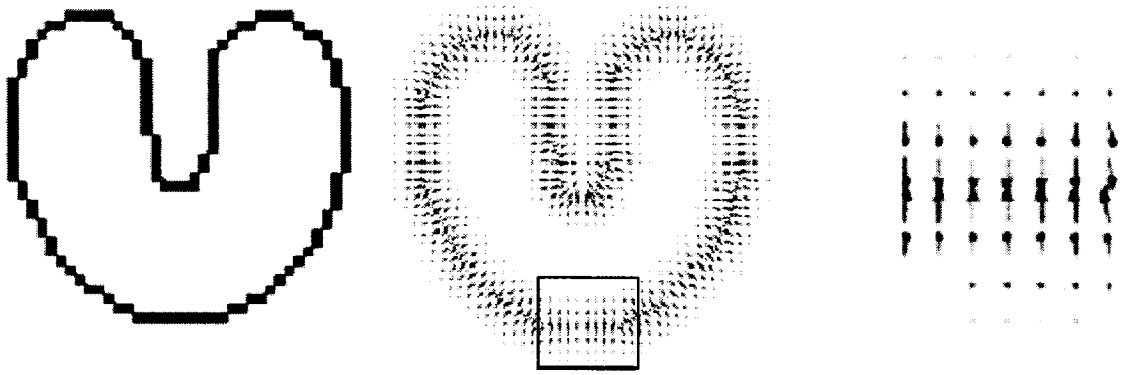


Image External force Zoomed in

Figure 1. 6 Example of external force acting on the contour (source: [6])

As seen from the figure 1.6 above, the external force acting on points near the boundary tries to pull the points towards the boundary.

1.5.6 Gradient vector flow

Application of the traditional snake model has had limitations because of two main drawbacks to the model. First it has a small capture range. The image gradient based external force dies out quickly away from the center of edges. An example of that phenomenon is shown in Figure 3.4. As seen in the figure, the external force only acts in the immediate vicinity of the boundary. Since the gradient of image is very low and essentially 0 for smooth areas, the external force acting on a control point in this region is 0. Therefore snakes have a very small capture range and they should be initialized close to the object boundary we want to detect. This is a very demanding condition. A solution to this problem is to apply a very strong Gaussian low pass filter to smooth out the edges, but by doing this we may have less accurate results and could miss the concave regions completely. The second problem of traditional snake model is that it fails to detect boundary concavities. This is still because of the external force. Figure 1.8 shows a zoom in of the concave region in Figure 1.7. It can be seen that the external force acting on a

control point in the concave region simply pulls it in the horizontal direction. There is no force acting down pulling a point further into the concave region towards the inner edge. So the contour cannot reach inside the concave region of the U-shaped object.

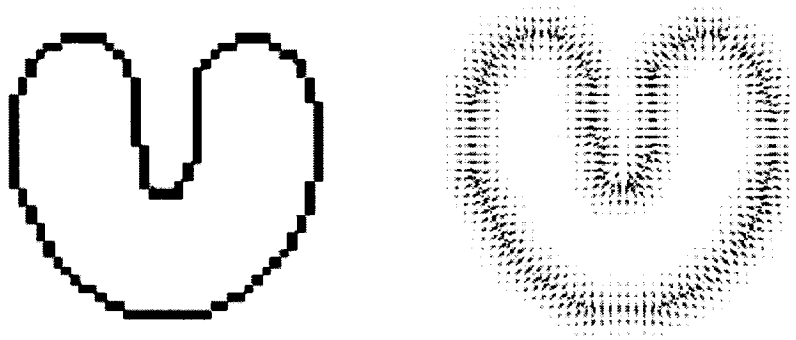
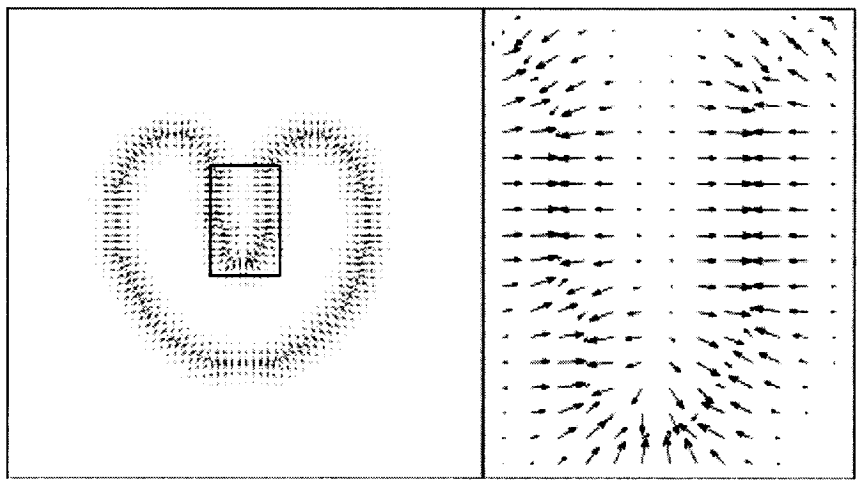


Image External force field of the image

Figure 1. 7 Example of traditional external force field (source: [6])



Traditional force field Zoomed into the marked region

Figure 1. 8 Example of external force in concave region (source: [6])

The technique presented by Xu and Prince [6] addresses these issues and presents a new formulation for active contour modeling. They defined a new external force field known as Gradient vector flow field and hence the name GVF snake. The basic model for GVF snake is the same as traditional snake. However GVF field is much better than the

image gradient based external force field for many applications and overcomes the two problems mentioned above. Some other approaches were also proposed to solve the two problems, but they still had some limitations, a detailed analysis is presented in [69]. In this work, capture range is of concern, because a greater capture range allows a bigger movement of the contour to be captured. As for the ability of reaching the boundary concavities, it is not necessary a desired feature in some situations as demonstrated in section 3.3. We describe the GVF snake model in this section using the notations and mathematical derivations originating from [6] and [69].

1.5.6.1 Model of GVF snake:

The GVF snake is defined as a contour $\mathbf{v}(s) = (x(s), y(s))$ which satisfies the following Euler equation

$$\alpha v_{ss} - \beta v_{ssss} + V = 0 \qquad \text{(Equation 1.22)}$$

where $V(x, y) = (u(x, y), v(x, y))$ is the vector field which substitutes the image gradient based external force field in the traditional snake model. The internal forces are defined similar to the original model consisting of elastic and bending forces. The GVF field $V(x, y)$ is the key here.

The formation of GVF field starts by calculating the edge map of the given image, using any edge definition such as

$$f(x, y) = |\nabla \mathrm{I}(x, y)|^2 \qquad \text{(Equation 1.23)}$$

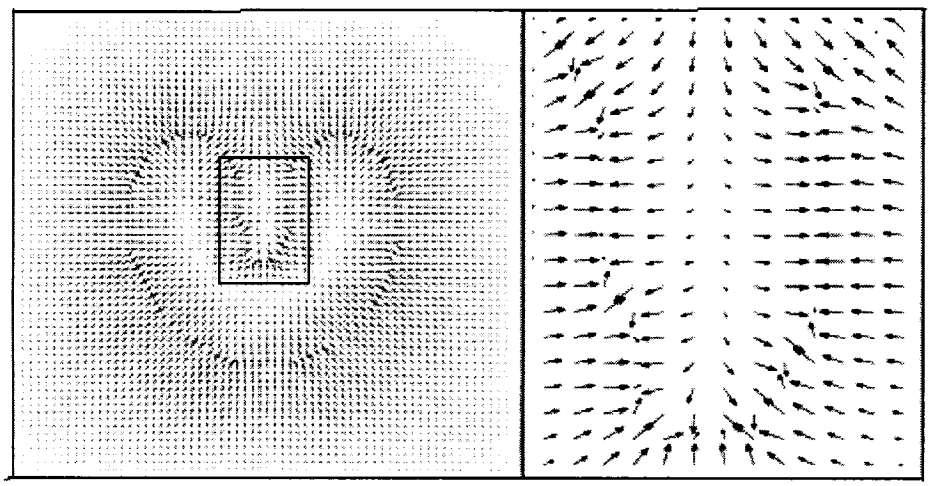
$$f(x, y) = |\nabla (G_\sigma(x, y) * I(x, y))|^2 \qquad \text{(Equation 1.24)}$$

The edge map has three important features relating to snakes deformation. First, in the edge map, there are vectors pointing towards the edges, which is a desirable property for snakes. Secondly, these vectors have large magnitude at the vicinity of the edges. Finally, in homogenous regions (regions with little variation in image intensity) the magnitude of vectors is almost zero. The second and third features can be problematic when constructing an active contour. To keep the first feature and nullify the effect of the latter two, the gradient map is extended farther away from the edges and into homogenous regions using a computational diffusion process.

The gradient vector flow field is defined as the vector field $V(x, y) = (u(x,y), v(x,y))$ that minimizes the following energy functional:

$$E = \iint \mu(u_x^2 + u_y^2 + v_x^2 + v_y^2) + |\nabla f|^2 |V - \nabla f|^2 \, dxdy \qquad \text{(Equation 1.25)}$$

where u_x, u_y, v_x, v_y are the partial derivatives of $u(x,y)$ and $v(x,y)$ in the x and y directions. The parameter μ is a regularizing parameter which adjusts the tradeoffs between the first and second terms of the integrand and is set according to the level of noise present in the image [6]. As can be seen, when the value of the edge gradient is small, energy is dominated by the sum of the partial derivatives of the gradient field. When the gradient is large, the second term dominates. In this case, setting $V = \nabla f$ minimizes the energy. Overall, this formulation transforms the gradient vector flow field by keeping it equal to the edge gradient at the boundaries; it also keeps V slowly varying at the homogenous regions of the image. Figure 1.9 gives an example of GVF field. Note the large capture range and the force acting in the concave region. This is not the case with traditional snakes. In fact a GVF snake can even be initialized far from the object boundaries.



GVF force field Zoomed into concave region.

Figure 1. 9 Example of GVF field (source: [6])

1.5.6.2 Numerical solution to the model

Finding a vector field which minimizes E is a variational calculus problem and it can be solved by applying Euler-Lagrange differential equations [6]:

$$\begin{aligned} \mu\nabla^2 u - (u - f_x)(f_x^2 + f_y^2) = 0 \\ \mu\nabla^2 v - (v - f_y)(f_x^2 + f_y^2) = 0 \end{aligned} \qquad \text{(Equation 1.26)}$$

Here, ∇^2 is the Laplacian operator. The Euler equation can be solved numerically by treating u and v as a function of time t. The resulting equations are:

$$\begin{aligned} u_t(x,y,t) = \mu\nabla^2 u(x,y,t) - (u(x,y,t) - f_x(x,y))(f_x(x,y)^2 + f_y(x,y)^2) \\ v_t(x,y,t) = \mu\nabla^2 v(x,y,t) - (v(x,y,t) - f_y(x,y))(f_x(x,y)^2 + f_y(x,y)^2) \end{aligned} \qquad \text{(Equation 1.27)}$$

The steady-state solution of Equation 1.27 yields the solution to the Euler equations in Equation 1.26. An iterative solution can be set up for solving the equations.

To solve the problem in image domain, everything needs to be discretized. In the discrete model, the u and v components of the GVF field are defined for each pixel of the image. Therefore $u(x, y)$ and $v(x, y)$ are represented by two matrix of size equal to the image. The above Euler equations are applied to each pixel in the image repeatedly. The parameter μ is taken from input and is kept constant throughout the process. The Laplacian operation is performed by applying the following mask.

0	1	0
1	-4	1
0	1	0

The vector field finally obtained is normalized to have a unity magnitude. The discretization of other forces is the same as the traditional snake model.

Chapter 2 Methodology

Endoscopic surgical images differ from natural scenes in that they are acquired in a compact viewing area with limited illumination, and are mainly composed of tissues in similar red-like colours. Another difficulty of this kind of images is specular reflections from moist tissues and metallic instruments that change unpredictably from frame to frame. The nature of this kind of images (noisy, low contrast and with fuzzy boundary) requires image pre-processing, such as smoothing, contrast enhancement, etc. Pre-processing can enhance discriminating features, but it is still hard to obtain a perfect segmentation from graph-based method, especially with endoscopic images. In fact, endoscopic images are hard to segment compared to many other kinds of images we have seen in the literature.

Recall that our first subtask is to find the precise boundary of the cavity in a frame, and the resulting contour will be used as the initial contour of snake tracking. We find the initial contour by a dedicated segmentation method. The only human intervention required is a mouse click in the resulting segmentation map to identify the cavity region that is going to be tracked. The proposed segmentation method that integrates graph-based segmentation and region merging is composed of the following steps:

1) Pre-process input images. Processing includes Gaussian smoothing, brightness and contrast enhancement, colour space conversion and specular reflections removal.
2) Do a relatively coarse graph-based segmentation. The parameter that defines the stop condition can be roughly selected because it will not affect much the final segmentation result. The impact of the parameter will be discussed later.
3) Do a simple post-processing to remove very small regions. These regions are considered to be spurious areas.
4) Do a multistage region merging with different criteria at different stages. The criteria are selected by making use of prior knowledge or reasonable hypotheses that will be described later. We use a formula to combine different criteria for computing a merging score, and then decide whether to merge or not by a threshold.

These steps are shown in a flow chart (Figure 2.1), and are described in detail in the subsequent sections.

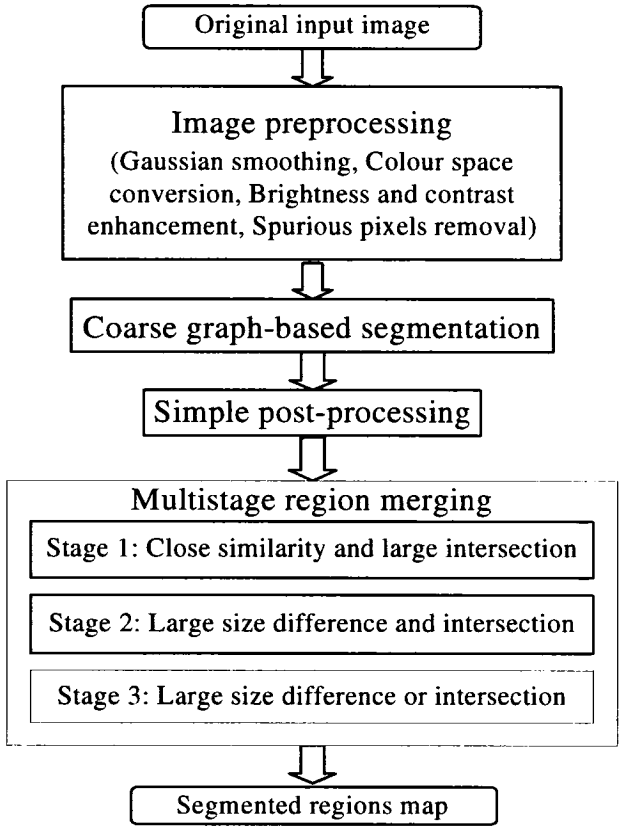


Figure 2. 1 Flow chart of the proposed segmentation method

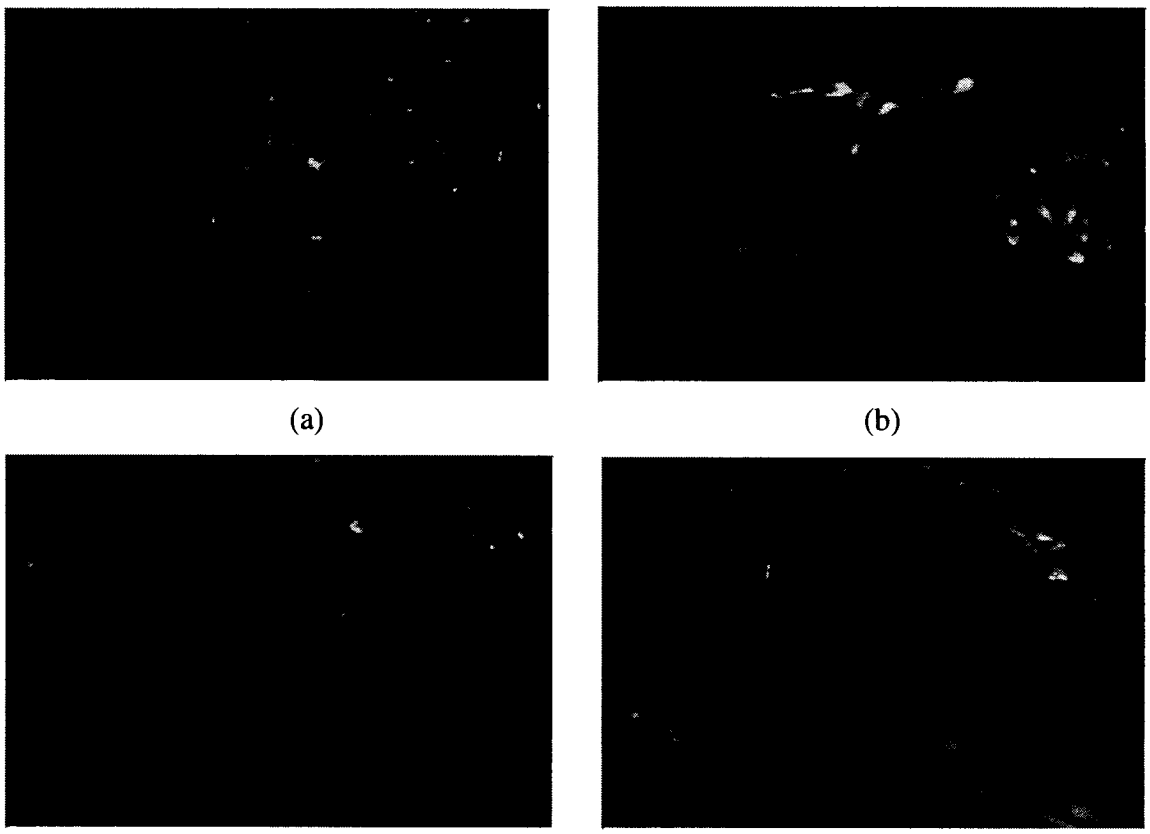
After identifying the boundary of the cavity using the dedicated segmentation method, we need to track the cavity area in the successive frames. In the tracking phase, first the image pre-processing described in section 2.2 is applied on current frame, and the gradient vector flow (GVF) is computed from the pre-processed image. The contour from previous frame or specified by the user is then evolved to the new boundary in current frame through the energy minimizing process of traditional snake. Next, the new boundary is deformed again in the guidance of GVF snake. The result of GVF snake is interpolated to form a dense contour, which is used as the initial contour of next frame. Therefore the original contour is tracked from frame to frame as the process continues.

This chapter is organized as follows. First we describe our consideration on feature selection in section 2.1. Then we discuss the image pre-processing methods specially tailored for our endoscopic images in section 2.2. Section 2.3 deals with the adopted graph-based segmentation algorithm. Its theory has been introduced in section 1.4; here we describe how it is used and adapted for our application. In section 2.4, we discuss the drawbacks of the graph-based segmentation algorithm and present our solution (a multistage region merging algorithm) in detail. In the last section, we present how the combined traditional snake and GVF snake is used in the tracking of cavity area.

2.1 Feature selection

Segmentation algorithms are based on some image features to define homogeneous regions. Many image features can be used as criteria of homogeneity for image segmentation, such as colour, intensity (grey level), texture, motion vector, elemental model (e.g., shape model), etc. The features best for a segmentation task depend on the algorithm used as well as the specific application data. Thresholding, edge-based and region-based methods often use single feature and constant distance measure, while clustering and statistical methods often use multiple features with a weight for each feature, and the weights are dynamically adjustable through a feedback loop so the distance measure is also dynamic. Texture and motion information are reliable in highly textured images, while colour and intensity information are more efficient in relatively uniform images.

For endoscopic surgical images, due to noise, poor illumination and similarity in colours, boundaries between different structures are not clear. There are also some unwanted specular reflections in the images. Figure 2.2 shows four image frames (non-consecutive) extracted from the endoscopic video sequence. We use them as examples in discussion of feature selection.



(a) (b)

(c) (d)

Figure 2. 2 Sample image frames from the endoscopic video sequence. (a) frame 1, (b) frame 60, (c) frame 120, (d) frame 460.

Now let us see what features are helpful for us to identify different regions in Figure 2.2 starting with a general observation on human perception. Intuitively, colour, texture and shape are the features we use most often to differentiate objects [31]. By using these three features, we can often tell the differences among different or same type of objects, e.g., cloths. Based on this observation, many segmentation algorithms use colour, texture and shape features individually or conjointly. Some of these algorithms really work well in natural scenes. In our case, the images we are interested in are not so colourful; actually the whole images are full of similar red-like colours. We also cannot find much texture (or some kind of repeatable pattern) in the images. As for the shape, we can see from Figure 2.2 that the shapes of tissues are irregular and change from frame to frame, which result in the change of shapes of other regions. Hence, colors, shapes and

textures, which enhance recognition and distinction between objects, cannot be used. Even with these hurdles for our segmentation problem, different image regions still can be distinguished. Why? It is the pixel intensity computed from colour information that plays an important role. Colour is the measure of amplitude in terms of luminance or chrominance of the image. Since whole image pixels are variations of red, the chrominance does not help much. Whereas the luminance difference between tissues and other regions is actually large enough. This is demonstrated in Figure 2.3, where the colour images in Figure 2.2 are converted to intensity images. We will find that the intensity images are easier to be segmented than the colour images in this case.

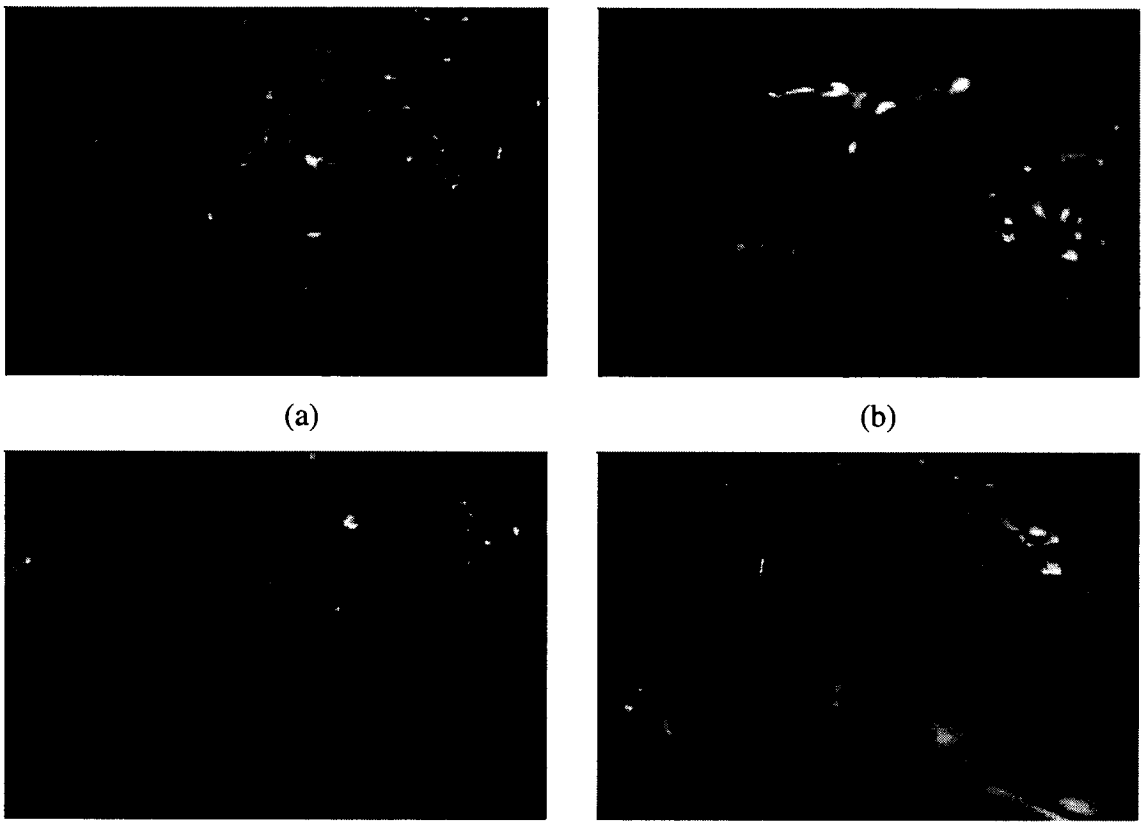


Figure 2. 3 Sample intensity images corresponding to those of Figure 2.2

Motion information is not so intuitive, even tough it is based on colours and textures. In image processing, it is often described by optical flow (see section 1.2.1).

Because motion information is an important cue for detecting moving objects in video sequences, we spent a lot of time on it, though we did not find useful motion data and gave it up at the end because of lack of textures which lead to incoherent and unreliable motion detection. There exist different algorithms to compute the optical flow; we have tried different versions from the classical Horn and Schunck's version [52], Lucas and Kanade's version [53], to a more recently proposed version by Black and Anandan [54]. Figure 2.4 gives an example of two consecutive frames and Figure 2.5, Figure 2.6 and Figure 2.7 give x and y components of the optical flows computed from the two frames using the three above mentioned optical flow algorithms. We used the implementation in OpenCV library [70] for the first two algorithms, while the third one was supplied by Professor Black himself.

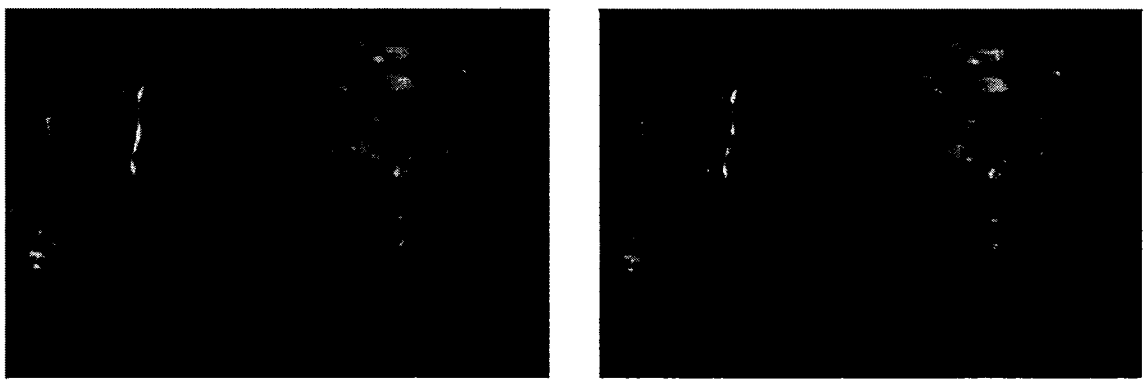


Figure 2. 4 Two consecutive frames for computing of optical flow.

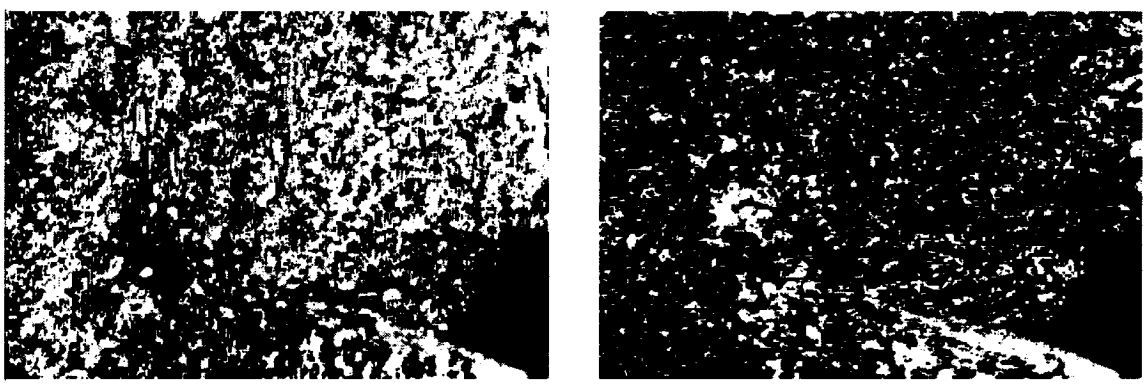


Figure 2. 5 Optical flow computed by Horn and Schunck's algorithm, left: x component, right: y component.

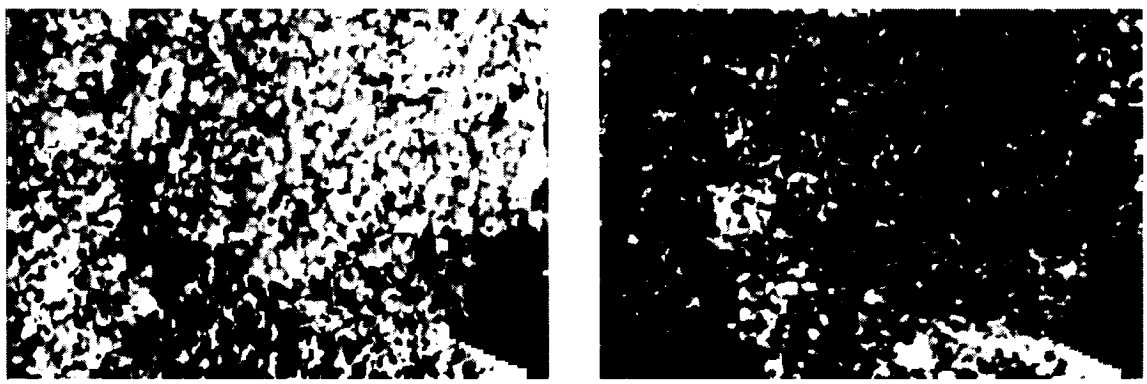


Figure 2. 6 Optical flow computed by Lucas and Kanade's algorithm, left: x component, right: y component.

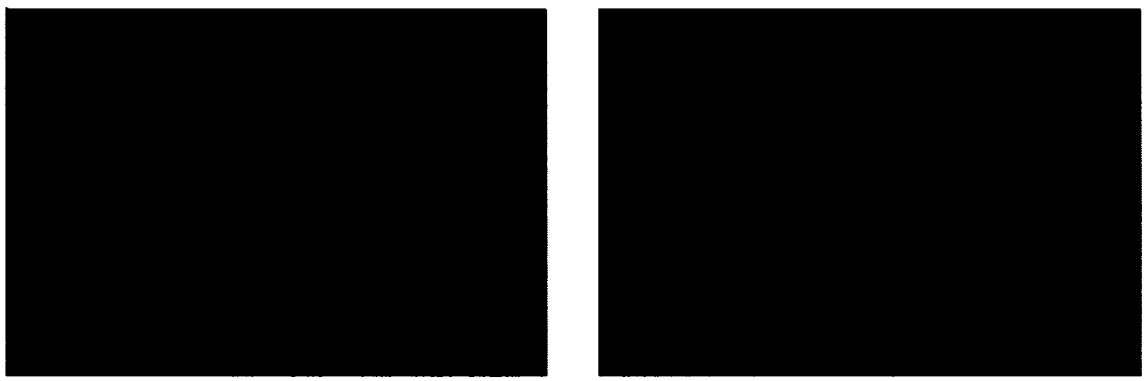


Figure 2. 7 Optical flow computed by Black and Anandan's algorithm, left: x component, right: y component.

The x (horizontal) and y (vertical) components of optical flow are originally float points; we normalized them within normal grey level range [0, 255] for display. So basically these figures (Figure 2.5 – Figure 2.7) show the magnitude of optical flow and the value can be directly used in a segmentation algorithm as other image features, e.g., intensity. The larger movement at a position, the bigger value the pixel at that position takes. So we should see brighter regions corresponding to moving objects in the optical flow images. In Figure 2.5 and Figure 2.6, the instrument is almost differentiable from the rest, although it is incomplete (see the lower part of the images); but the cavity or other tissues are almost noises appearing in the images. Figure 2.7 hardly gives useful

information that we are interested in. To see what we can get from motion information, we also tried to use the optical flow data in an actual segmentation process. We experimented to use the Manhattan distance computed from x and y components of optical flow as the weight of edge in graph-based segmentation algorithm (see section 2.3), and the results were quite disappointing. Looking at the images in Figure 2.2 and Figure 2.4, we may observe that there is no much texture and no dominant background or foreground in endoscopic images. This is possibly the reason that we cannot obtain good motion data thus cannot do a good segmentation with motion data.

Colour, intensity, texture, shape and motion information are the most often used features in image/video segmentation. Some researchers also used space information (i.e., coordinates of pixels) in their clustering segmentation methods [34, 35]. It is helpful to add spatial influence during clustering, but meaningless for region-based especially graph-based methods, because they incorporate spatial data naturally in their segmentation process: the nodes (pixels or regions) in a graph are always merged with neighbouring ones. Now we have seen that there is not much texture and there is no dominant background or foreground in endoscopic images, so texture and motion vector are not reliable in this situation. On the other hand, the colour differences among regions are small, and the shapes of the regions are variable, so colour and shape model are also not reliable criteria. In fact, intensity is the most commonly used criterion, because it is a fundamental characteristic of images, and it is simple and efficient in many situations. In our case, it allows distinguishing the reddish regions. We will use intensity as basic criterion during segmentation and do some pre-processing to make the region boundary clearer before applying the segmentation algorithm.

2.2 Image pre-processing

Images are basically a set of pixels that are often less than a perfect representation of reality. By pre-processing some unwanted variations, noise can be reduced and desired features enhanced [8]. We apply the following procedures to pre-process our data.

- Gaussian smoothing
- Brightness and contrast enhancement

- Colour to intensity image conversion
- Specular reflections removal

Gaussian smoothing allows us to remove variations and noise. In this work, we used a 2-D convolution operator with a standard 5 x 5 kernel. The 2-D Gaussian (bell-shaped) kernel has a smoothing effect on the image since it is a low-pass filter and has the characteristic of averaging among the neighbouring pixels within an image. By smoothing an image, the Gaussian removes noise. The smoothing actually blurs the image especially the sharp edges. Other kernels such as 3 x 3 or 7 x 7 also can smooth the image, but we found in experiment that the 5 x 5 kernel is more suited to our data as it filters noise and blur the image moderately.

Enhancing brightness and contrast allows us to improve information visibility. We used the classic brightness/contrast adjustment algorithm. The implementation is described in [74], in which brightness was implemented as a simple constant added to or subtracted from each pixel in the image; contrast was a saturated multiplication by a contrast adjustment factor c which darkened with intensities below 128 and brightened those above it. Specifically, the piecewise function applied was:

$$f(x)=\begin{cases}0, \text{ if } (x-128)\times c+128\le 0\\ 255, \text{ if } (x-128)\times c+128\ge 255\\ (x-128)\times c+128, \text{ otherwise}\end{cases} \qquad \text{(Equation 2.1)}$$

After smoothing and enhancement on the colour image, the image is then converted to intensity image used in the segmentation algorithm. There are many ways to do the colour space conversion [55]. The following three standards are most often used for RGB to grey level conversion:

- Rec. 601-1: Intensity = 0.299 * Red + 0.587 * Green + 0.114 * Blue
- Rec. 709: Intensity = 0.213 * Red + 0.715 * Green + 0.072 * Blue
- ITU standard: Intensity = 0.222 * Red + 0.707 * Green + 0.071 * Blue

We have chosen to use the Rec. 709 standard because it gives less weight to the red component, and hence increases the differences between the reddish regions.

Finally, pre-processing ends by removing spurious pixels that are actually very high intensity pixels caused by specular reflections from polished instruments and liquids in the scene. We analyse image histogram to find those specular reflections, then used simple thresholding to truncate the intensity of these pixels. This is very important, because those pixels have large weights and they may affect region-growing result unexpectedly. We will give examples of results for each step of the endoscopic image pre-processing in Chapter 3.

2.3 Graph-based segmentation implementation remarks

We adopted the efficient graph-based segmentation algorithm [5] as described in section 1.4. The graph describing an image can be 4-connected or 8-connected. In our case, we have chosen 4-connected graph because it forms fewer edges so that the process is faster. On the other hand, it will result in a coarser segmentation that is just what we need for the multistage region merging process. We want a coarse segmentation from the efficient algorithm because we found experimentally that a fine segmentation could mix up the cavity area with other parts in endoscopic images, but we can use the result of coarse segmentation in the following multistage region merging process, which employs some application specific information, to get adapted results. Some experimental results are given in Chapter 3. Note the efficient algorithm proposed in [5] is a good algorithm, even if it approximates RSST and cannot be used alone in our application. Its efficiency has been proved in our experiment. In addition, several papers have referred to this algorithm, and some very good results from the algorithm were presented in [5] and other papers. It just does not work well, used alone, for our particular endoscopic images, but we can further process the coarse result from the segmentation algorithm and fulfill our specific requirements (refer to the multistage region merging process described in section 2.4).

Basically the RSST segmentation algorithm is conducted by recursively finding the least weight edge and merging the two regions (which are originally pixels) connected by the edge. After each merge, the weights of affected regions are recomputed. Each node and each edge has a corresponding weight. The weight of a node is the intensity (grey level) in our case, and the weight of an edge is the weight difference of the

two ends. During the segmentation process, similar neighbouring pixels are grouped into regions, and then the weight of a region is simply the average intensity of composing pixels. As described in section 1.5.2, the conventional RSST based segmentation algorithm [58] will recompute weights of related edges as well and then find the least weight edge to merge. Since sorting of edges is slow as there are a lot of edges (e.g. 168 368 edges in a 352 x 240 image), putting it in the computation cycle is really time-consuming. The algorithm described in section 1.5.3 is more efficient, because it neither recomputes weight of edges, nor re-sorts edges, but uses an internal difference criterion to evaluate if two regions can be merged. Although the efficient algorithm makes the segmentation criterion and stop condition not as clear as the traditional one, it is much faster (nearly real-time running on a Pentium IV 2.4GHz PC). We found the efficient algorithm was hundreds of times faster than the conventional one on segmenting a typical 352 x 240 image, and the quality of segmentation result is not much different. Actually, the exact level of similarity depends on the parameters used. In our case, it does not matter since we just need a coarse result from graph-based segmentation. We use this efficient algorithm to generate an initial segmentation map as the input to the multistage region merging process.

The authors of the efficient RSST implementation gratefully supplied their source code for research. We tested it in our application and made two modifications to make it more efficient for our application. One important modification is the sort algorithm. It is critical from efficiency point of view. The original sort algorithm is based on the "quick sort" algorithm. Our experiment shows it is not very efficient in sorting large amount of data. We then changed it to the "heap sort" algorithm and the efficiency improved dramatically. Another modification is the graph data structure. The original structure is an 8-connected graph; we changed it to 4-connected. We also modified the edges data structure to allow processing of floating point input data from two images, so that we could test the segmentation based on the optical flow data which are composed of two components.

2.4 Multistage region merging

Although graph-based segmentation methods like RSST and the variations are very good bottom-up approaches, they do not generate acceptable results all the time. That is because they have some intrinsic drawbacks. As mentioned above, graph-based segmentation relies solely on weight differences as merging criterion, and it will merge low weight difference regions iteratively. Normally, the algorithm will not stop merging until a certain number of regions are left [64]. A threshold for the edge weight can be used as a stop condition also. In some other graph-based methods such as [5], an initial weight is defined and it will ultimately determine when to stop merging. We call the parameter that defines the stop condition K. We observe that there are three main drawbacks to graph-based segmentation:

- The optimal K is hard to determine, and inappropriate K can cause under-segmentation or over-segmentation. In fact, the value of K depends on the image characteristics of the specific application.
- The average intensity among regions tends to be closer when the regions grow bigger. An average intensity difference criterion may be reasonable at the beginning, but it may not be the most appropriate one when regions grow to some extend.
- Prior knowledge is not used in the merging procedure, so the result is hardly controllable. For example, we expect closed regions in many cases, but regions in the acquired image may not be perfectly closed due to noise, and it is quite possible that the algorithm merges some regions with other regions outside of the actually closed area. Another example is small regions remaining inside bigger ones up to the end, because the weight difference among the small regions and their neighbours are too large. Those small regions are possibly spurious areas that we want to get rid of.

The output of any segmentation method can be improved by simply merging similar neighbouring regions together. The idea of region merging is similar to that of graph-based method, but it starts from regions instead of pixels, and can use different

merging criteria at different iterations. Merging order and merging criteria are fundamentals of merging algorithms. The merging order defines the order in which the region links should be processed to determine the sequence of merging. The merging criteria decide whether the two regions should be merged or not, and usually they are functions of some properties of the two candidate regions to be merged.

There are a lot of region merging algorithms proposed in the literature [66, 67, 68, etc.]. Compared to these methods, our algorithm is composed of three merging stages and it uses different merging criteria. At each stage, regions are merged iteratively from small to large until convergence, i.e. no more merging. Merging scores are computed and thresholded at each stage. Stop conditions are used to prevent over-merging. The merging scores are computed from the following region features: grey-level similarity, region size and common edge length. The formulation of the scores is based on what we believe are reasonable hypotheses in the context of our work, and it is distinct in different stage as we will see in the following subsections. The stop conditions are based on prior knowledge; in our case, we use minimum number of regions and minimum and maximum size of regions (percentage of the whole image). The stop conditions are kept unchanged at each stage. Note our multistage region merging algorithm is specially designed for a fine segmentation of endoscopic images; it is original and have contributed to the publication of a conference paper [73].

2.4.1 Merge stage one

Discrimination between adjacent areas with different means and standard deviations can be made according to Fisher's criterion (the ratio of the between-class variance to the within-class variance) [10]:

$$F = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2}, \qquad \text{(Equation 2.2)}$$

Where μ and σ^2 are the mean and the variance operators respectively. In other words, if two regions have good separation in their means, and low variance, then they can be discriminated. However, if the variance becomes high and the mean difference is low it is not possible to separate them.

Although Fisher's criterion is a good representation of similarity, we also want the two neighbouring regions that are going to be merged to be intersecting each other as much as possible. By intersecting, we mean that they have some boundary pixels side by side and these pixels form what we call a common edge. In practice, we also found that a variation of Fisher's criterion has better performance in our algorithm. We write it as F':

$$F' = \frac{|\mu_1 - \mu_2|}{\sqrt{\sigma_1^4 + \sigma_2^4}}. \qquad \text{(Equation 2.3)}$$

So we have the merging score formula:

$$\text{Score 1} = \frac{|\mu_1 - \mu_2|}{\sqrt{\sigma_1^4 + \sigma_2^4}} \times \frac{P}{L} \qquad \text{(Equation 2.4)}$$

Where P is the perimeter of the smaller region and L is the common edge length, P and L are counted in number of pixels. Note F' gives less emphasis on mean difference than F, which is consistent with our data, because the blood spill changes the intensity of nearby area in surgical images and difference in the mean must not be over-emphasized or else region are not merged together very well.

Given a region, we will find the smallest score among its neighbours, and then merge the two regions if the score is less than a threshold (determined through experiments) and the common edge is not too small ((P/L)<5, means 20% of the perimeter of the smaller region). Hence at this stage, we favour merging regions of close similarity as well as with large intersection.

2.4.2 Merge stage two

Now that we have merged similar regions, suppose some ribbons of regions remain unmerged. Generally speaking, these ribbons are not likely to be independent objects. In our case, we want them to be merged into big neighbours as long as they intersect each other adequately. The merging score should include a size term and a common edge term:

$$\text{Score 2} = \frac{S_1}{S_2} \times \frac{P}{L} \qquad \text{(Equation 2.5)}$$

Where S_1 is the size of the smaller region, S_2 is the size of the larger region, P is the perimeter of the smaller region and L is the common edge length. S_1, S_2, P and L are counted in number of pixels.

Given a region, we will find the smallest score among its neighbours, and then merge the two regions if the score is less than a threshold (determined through experiments) and the intensity difference is moderate (<25). So at this stage, we favour merging of regions with large size differences as well as large intersection.

2.4.3 Merge stage three

This stage is a relaxation of stage two for further forming larger regions. First we disregard the size factor (S_1/S_2) and focus on the common edge term (P/L) to remove regions that are almost contained in other regions. Then, we disregard the common edge factor and focus on the size term to remove small regions adjacent to very large regions. At this stage intensity difference is still used as a condition to avoid merging too different regions, but it is relaxed too (<50).

2.4.4 Implementation remarks

After applying the three merging stages on the segmentation map (the output of graph-based segmentation), only several large regions are left. In many situations these regions correspond to meaningful objects in the endoscopic image, such as blocks of tissues, cavities and instruments. This result is usually not achievable for the graph-based segmentation algorithm, even if the parameter is especially tuned and a best parameter is found (by numerous experiments) for each image. We will show some experimental results in Chapter 3.

It is worth mentioning that the multistage region merging process is slow compared to the graph-based segmentation. That is mainly because it is a recursive process similar to RSST, and the computation of merging scores is time-consuming, especially on finding adjacent regions and common edges, even though we used efficient logical operators and mathematical morphological operators. At the beginning of this chapter, we have mentioned the use of a simple post-processing to remove very small regions after the graph-based segmentation. One reason for this is to reduce the number

of regions which in turn reduce the burden of the multistage region merging process. It is usual to find small regions remaining inside bigger ones up to the end for graph-based segmentation algorithms due to their intrinsic drawbacks. The simple post-processing finds very small regions in the segmentation map (e.g., size is less than 6 or about 0.01% of the entire image), and assigns the pixels in those small regions to the neighbouring regions with a scan order of top left to bottom right.

Currently multistage region merging process is not a real-time process, and some improvement may be made in the future if the real-time ability is required. In this work, we put more emphasis on the quality of segmentation result than efficiency, and in fact, on image sequences, we only apply the segmentation process when necessary. To be more precise, we do not need to segment each frame, but segment some specific frame to localize the cavity area as accurately as possible, and then we track the area in the consecutive frames using snakes as described in next chapter.

2.5 Application of snake model in cavity tracking

We have presented the theory of traditional snake and GVF snake in section 1.5.5 and section 1.5.6. Generally speaking, GVF snake performs better than traditional snake in object tracking applications, because the former has large capture range and is able to detect boundary concavities. In this work, we need the snake to track the cavity area in endoscopic video sequences. The cavity and surrounding tissues are all unstructured, and their boundaries are unclear because they intersect or occlude each other at some parts. Also noises or projected shadows in the captured image make some regions closer to the cavity area in terms of intensity value. So it is possible that the segmented cavity area includes some parts that are not really belonging to it, which is actually the area where inter-vertebral disc of the spine are located. Those undesired factors usually make the contour of segmented cavity area more irregular. In our experiment, we found that the segmented cavity often includes parts extruding into concave regions of neighbouring structures, which is not a desired phenomenon. The extruded parts will be kept by GVF snake, or even be developed further during contour deformation. To alleviate the influence of the undesired factors on tracking result, we first use a traditional snake to

regulate the contour and move it to the edges in close vicinity, then use GVF snake to further move the contour to strong edges in a larger range.

During implementation, we adopted the snake function in OpenCV library [70]. Basically this function implements the traditional snake model, with three parameters α, β and γ to adjust the weights of elastic energy, bending energy and external energy ($-\nabla E_{image}$); but constraint energy is not implemented. The algorithm for minimizing snake energy is different from the one based on variational calculus [3]. In fact, it adopted the greedy algorithm proposed by Williams and Shah [72]. This algorithm is more efficient and yields quite good results. The scheme of this algorithm for each snake point is as follows:

1) Use Equation 1.14 to compute E for every location from point neighbourhood. Before computing E, each energy term must be normalized using formula

$$E_{normalized} = \frac{E_x - \min}{\max - \min}$$

where E_x represents the three energy terms, max and min are maximal and minimal energy in scanned neighbourhood.

2) Choose location with minimum energy.
3) Move snakes point to this location.
4) Repeat all the steps until convergence is reached.

Criteria of convergence are as follows:

- maximum number of iterations is achieved;
- number of points moved at last iteration is less than given threshold.

So the function also requires parameters for defining the scanning neighbourhood (window size) as well as termination criteria. The same function is applicable for GVF snake, just by supplying the external force with GVF field. We used the C code on Xu and Prince's website [71] to compute the GVF field. Of course the function was modified

to accommodate the data structure in OpenCV. The snake function and GVF function are efficient and can run in real time on a Pentium IV 2.4GHz PC.

The contour is represented by a series of pixels in the image. In the beginning, the contour is sampled at each scan line to get a compact contour, whose discontinuity is not perceptible because there are no gaps between neighbouring pixels in sense of Manhattan distance. After snake deformation, the points forming the contour are moved and the contour may no longer be compact. So a resampling step is necessary to keep the contour compact during snake deformation. During the resampling step pixels are added or deleted depending on the length of the connecting edges. If the length exceeds a certain threshold a new point is added. If the length falls below a certain threshold a point is deleted. The threshold therefore controls the resolution of the model. We want the finest resolution, so a linear interpolation is applied for the resampling, and Manhattan distance is used to decide if there is a gap between two pixels.

Based on the deformable contour application model, we have designed a more efficient framework incorporating segmentation and tracking to fulfill our objective. The typical scenario of our application is described as follows. First a video sequence is loaded and a starting frame is selected. Then the system performs a dedicated segmentation as described in the former sections on that frame that will hopefully clearly distinguish the cavity from other structures. After that, the cavity area is simply specified by a click in the segmentation map, and then the system will go over the rest of the frames to track the specified area by means of snake methods. During contour tracking, the same pre-processing is applied on current frame, and the gradient vector flow is computed from the pre-processed image. The contour from previous frame or specified by the user is then evolved to the new boundary in current frame through the energy minimizing process of traditional snake. Next, the new boundary is deformed again in the guidance of GVF snake. The result of GVF snake is interpolated and used as the initial contour of next frame, so the original contour is tracked from frame to frame as the process continues.

Snakes do not perform perfectly all the time. When there is a significant change between two frames, such as the tracked object moves out of the view or is occluded by a surgical instrument, the snake may fail. In this situation, the contour may deform to fit another shape, which is not the object we are tracking. After that, the snake will deform according to the new shape and tracking of the original object is lost. It is meaningless to continue tracking in this situation. To solve the problem, we designed a function to detect a lost tracking event based on the average intensity difference between original contour area and the deformed contour area. When a lost tracking event occurs, the system will warn the user, and suggest him/her to restart the segmentation, object specification and tracking process from another start point (image frame). The following flow chart (Figure 2.8) depicts the complete application framework.

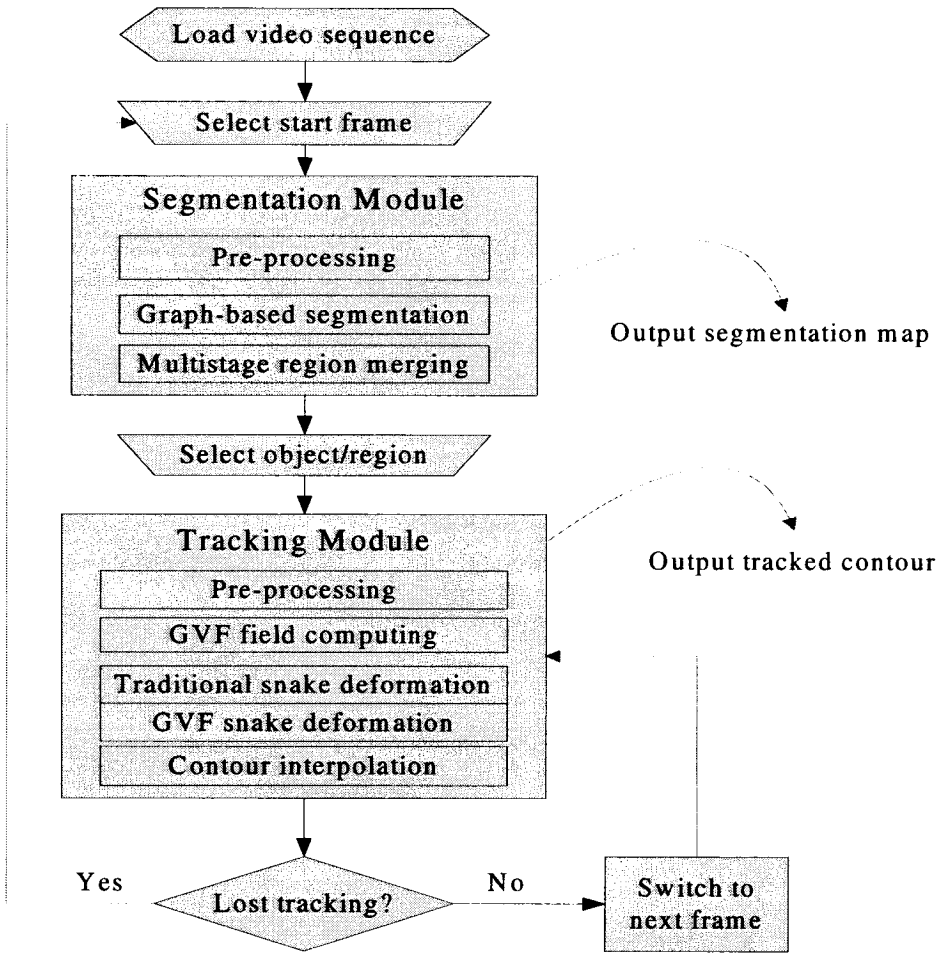


Figure 2. 8 Flow chart of the entire application framework

Chapter 3 Results and Discussion

Previous chapters have described the principles of pre-processing, segmentation and tracking methods used in our project. In this chapter we present some experiments validating each phase and discuss the effect of different parameters setup. The extracted pictures are generated by the application developed for this work. The application is implemented in C++, and it uses some image processing functions from an open source library (OpenCV, [70]). We also integrated the efficient graph-based segmentation algorithm [5] and GVF function [71] in the application. The original input of the application is an image file or a video file. A screen shot of our application is shown in Figure 3.1.

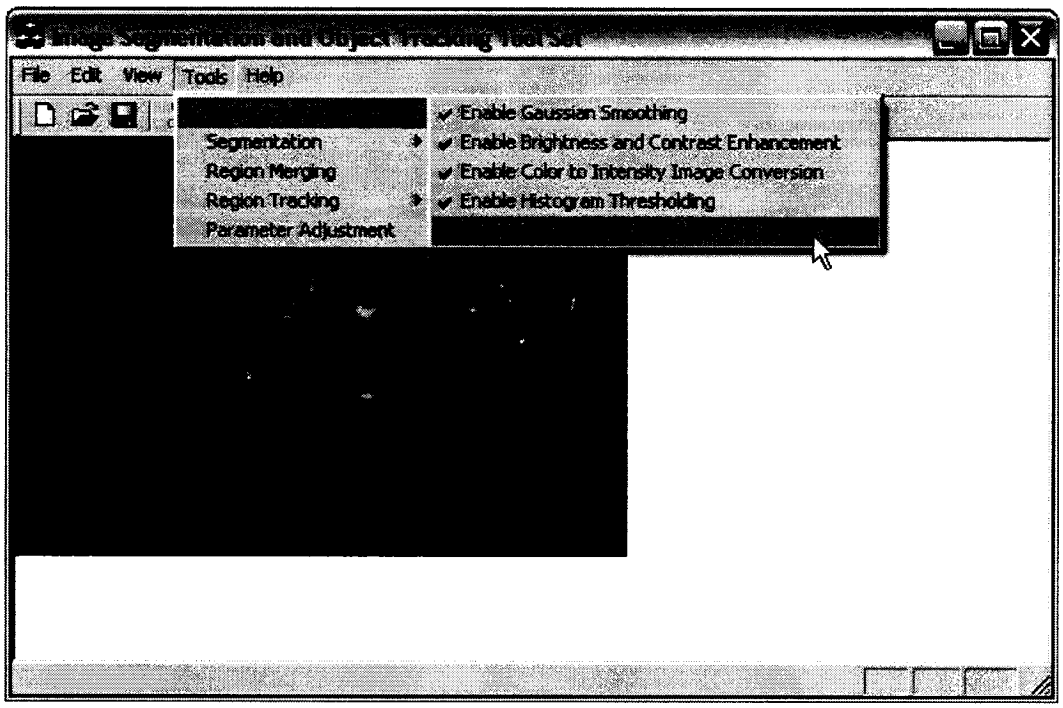


Figure 3. 1 Screen shot of our application

This chapter is organized as follows. Section 3.1 shows the result of individual pre-processing method as well as the compound effect of those methods. Section 3.2 compares the result of pure graph-based segmentation and the result of multistage region merging following a coarse graph-based segmentation. The final results are also

compared to the ground truth segmentation from a surgeon. Section 3.3 shows the contour evolution in consecutive frames by means of traditional snake and GVF snake. We also discuss some difficult situations such as false segmentation and false tracking in these sections.

3.1 Validation of pre-processing steps

To test the proposed pre-processing steps, we used two endoscopic video files (AVI format) supplied by Sainte-Justine Hospital in Montreal, as well as several other video files downloaded from the web. To be in accordance with our objectives, here we mainly report the experimental results obtained from an endoscopic surgical video file containing 850 frames. The AVI file is generated with a video compressor that reduces the information (not lossless) but the video is only altered mildly as the compression rate is not high. Image frames are dynamically extracted from the video file. Each frame is 352 x 240 256 colours. We test the frames with individual pre-processing step as well as the combined steps. The parameters used in each step have been evaluated experimentally, and some parameters yielding good results for a number of frames are kept to apply to other frames. Validation of pre-processing results is mainly qualitative, or by human perception. We also use the successive segmentation results as a feedback to evaluate the pre-processing parameters. Two original image frames (non-consecutive) are shown in Figure 3.2. The results of pre-processing shown in this section are based on these images.

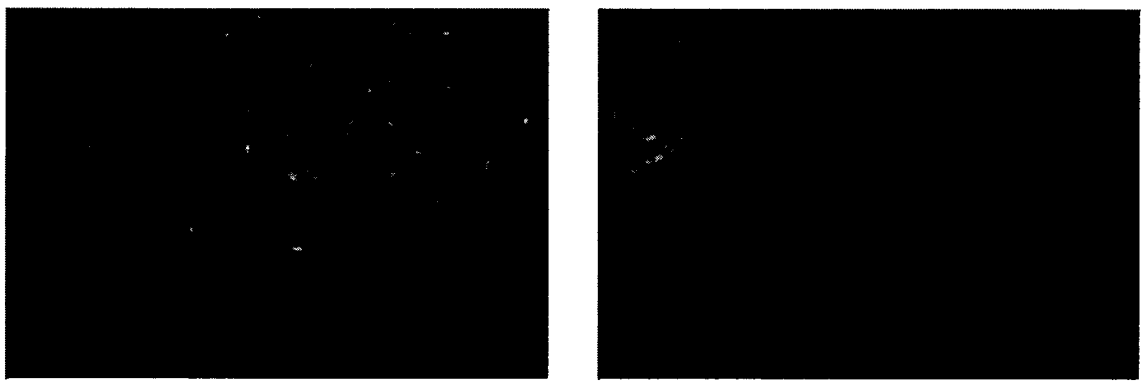
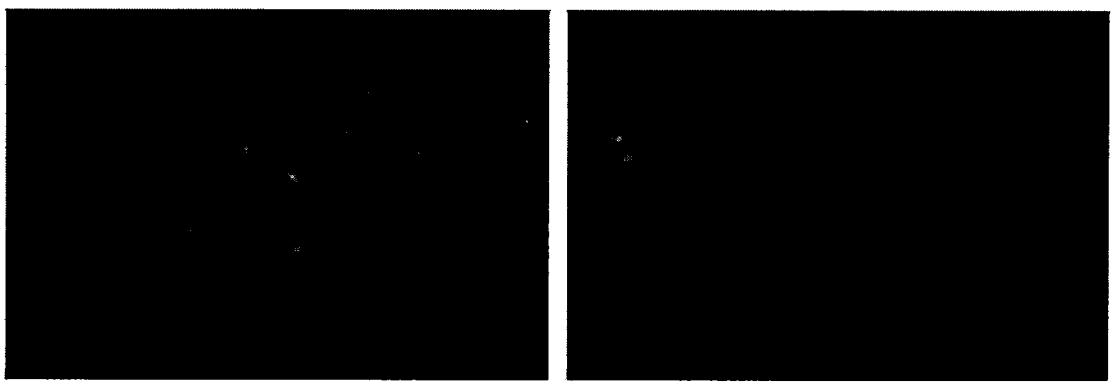


Figure 3. 2 Sample image frames from an endoscopic video sequence. Left: frame 8, right: frame 88.

3.1.1 Pre-processing results

Due to noise, low contrast and specular reflection, endoscopic images are hard to segment compared to many other kinds of images we have seen in the literature. Four pre-processing methods were adopted in our application; they are Gaussian smoothing, brightness and contrast enhancement, colour to intensity image conversion and specular reflections removal. This section illustrates the usefulness of these methods for obtaining the results discussed in section 3.2. They are applied serially on an image extracted from the endoscopic video sequence to enhance the discriminant feature (i.e., intensity) in the image so that the segmentation and tracking algorithm could yield better results. Each pre-processing method has some parameters to set up. In the following we first show the results of individual pre-processing with different parameters, and then we show the results of the combined pre-processing methods with the parameters that we think most suitable for our data.

The Gaussian smoothing function requires kernel size and sigma (standard deviation) as parameters. Using standard sigma for small kernels (3×3 to 7×7) gives better speed. Figure 3.3 gives the results of Gaussian smoothing with different kernel size applied on the images in Figure 3.2. From the results we can notice that Gaussian smoothing can generally smooth an image (or remove noises), but it also blurs edges. A larger kernel size results in a smoother image. We found from experiments that a moderate kernel size 5×5 yields a result suitable for subsequent processes in most situations, since it is a good compromise between noise removal and boundary clearness. So we use standard sigma and kernel size 5×5 in Gaussian smoothing.

(a) 

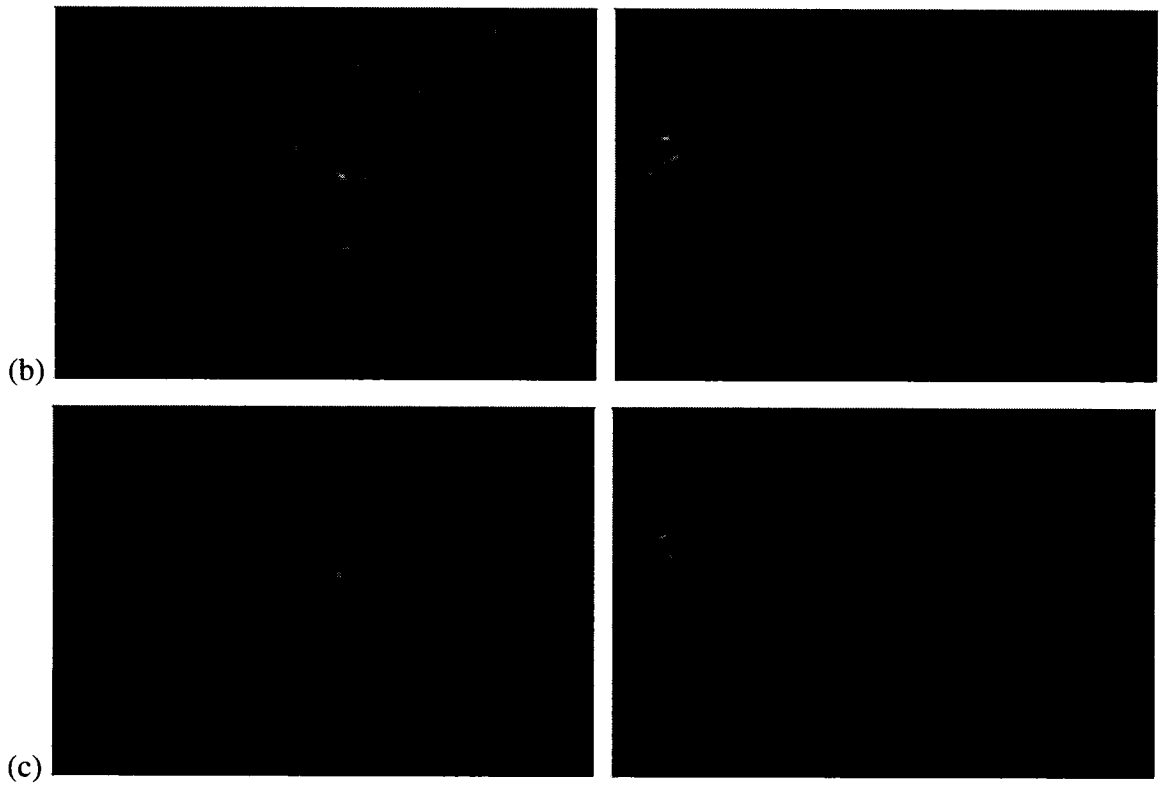


Figure 3. 3 Gaussian smoothing applied on the images in Figure 3.2. (a) kernel size 3×3, (b) kernel size 5×5, (c) kernel size 7×7.

The brightness and contrast enhancement function requires brightness and contrast values as parameters. These two values go from −100 to 100. A value less than 0 will reduce brightness/contrast, and a value greater than 0 will enhance brightness/contrast. The endoscopic images are poor illuminated, so we need to enhance brightness as well as contrast to increase the difference between different regions. There are many combinations with the two parameters; some combinations may yield similar results. We found from experiments that brightness 30 and contrast 70 yields a result suitable for subsequent processes in most situations. Figure 3.4 gives the results of brightness and contrast enhancement with brightness 30 and contrast 70 applied on the images in Figure 3.2. From the results we can conclude that this function can make the region boundaries clearer, and facilitates segmentation as it enhances strong edges and suppresses weak edges in the image.

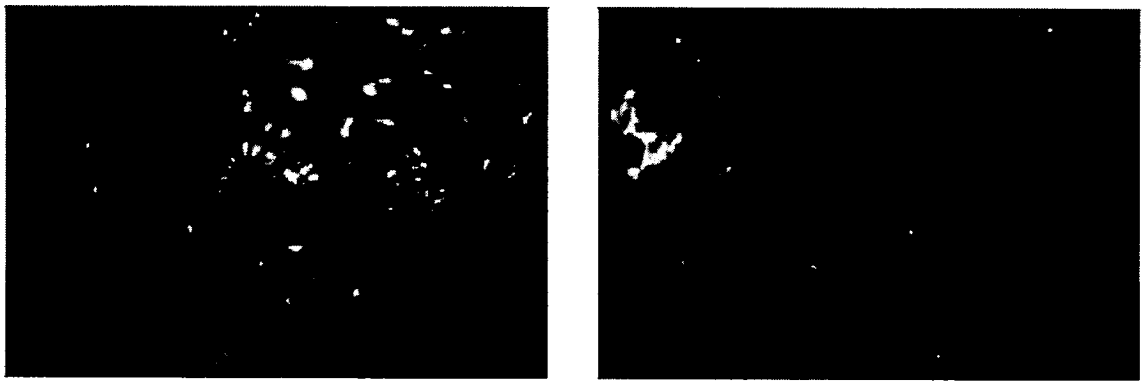
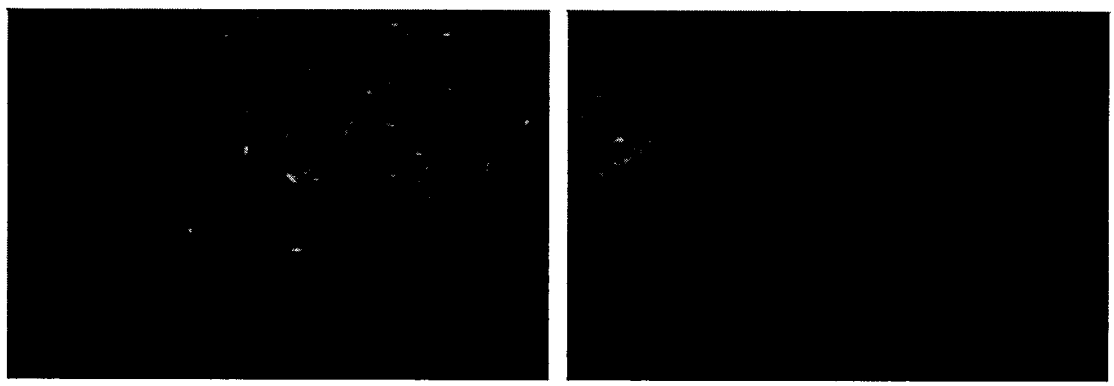


Figure 3. 4 Brightness and contrast enhancement applied on the images in Figure 3.2.

The colour to intensity image conversion function does not require any parameter, but there are different conversion standards (see section 2.2). Figure 3.5 gives the results of Rec. 601-1, Rec. 709 and ITU standard applied on the images in Figure 3.2. Although the differences are not very apparent to the human eye, converting images using Rec. 709 do work better in subsequent processes. A possible interpretation of this phenomenon is that the whole image pixels are variations of red, and Rec. 709 gives less weight to the red component, and hence increases the differences between the reddish regions. So we adopted Rec. 709 standard to convert colour images into intensity images. The segmentation and tracking algorithm all use the intensity images as input.

(a) 

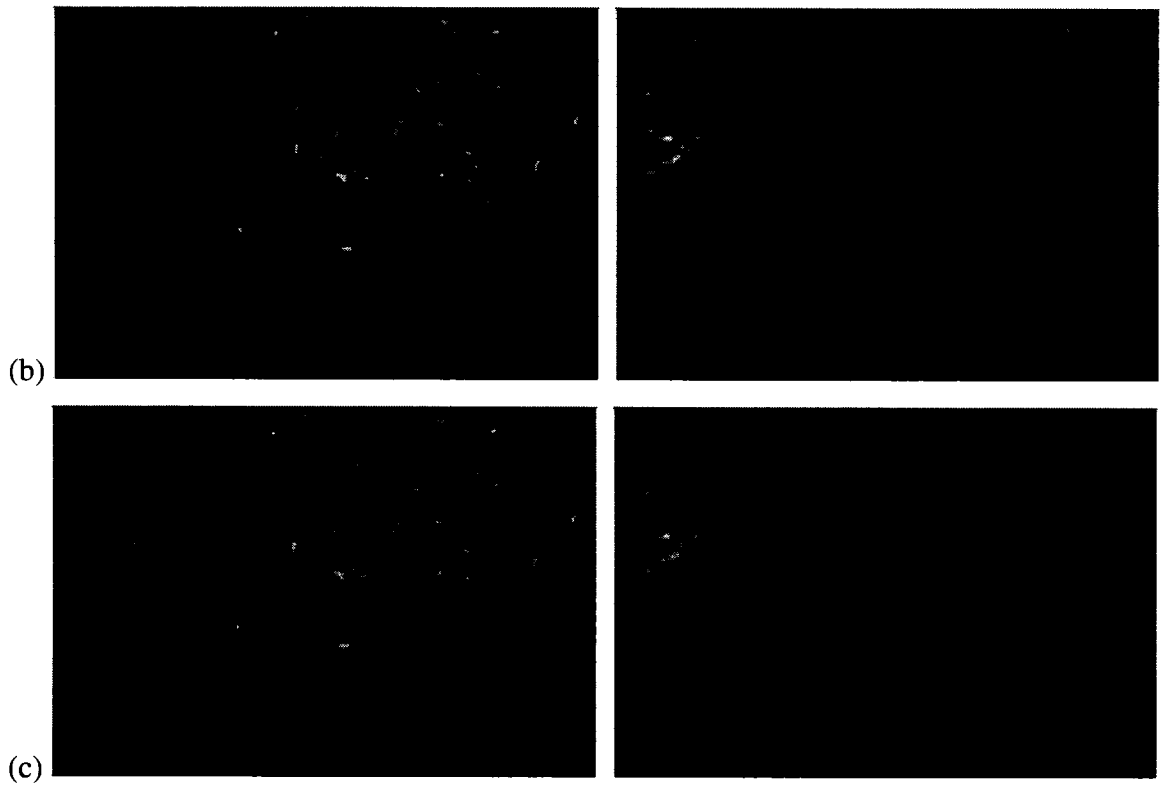


Figure 3. 5 Colour to intensity image conversion applied on the images in Figure 3.2. (a) Rec. 601-1, (b) Rec. 709, (c) ITU standard.

The histogram thresholding function does not require any parameter. It is applied on the intensity images. We use a simple thresholding to truncate the intensity values to the value corresponding to the maximum of the histogram. It is based on the observation that the tissues in the endoscopic image occupy a large portion of the whole image in most situations, and they have a large brightness (intensity) only in the surrounding of specular reflections. Hence, the specular reflections can be removed by a simple thresholding method. Figure 3.6 gives the results of histogram thresholding applied on the images in Figure 3.5b. This simple histogram thresholding may not work in more complex scenes. We are planning to integrate in the future the results from another researcher in our research group who is working on a sophisticated specular reflections removing algorithm.

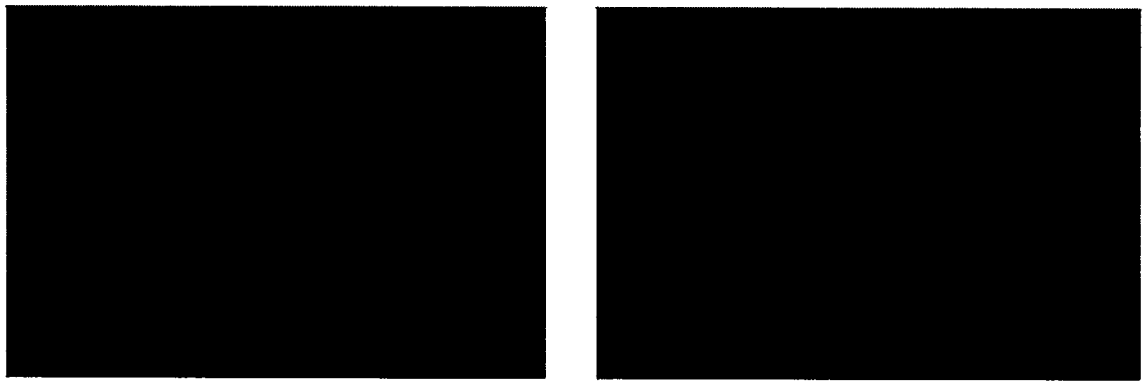
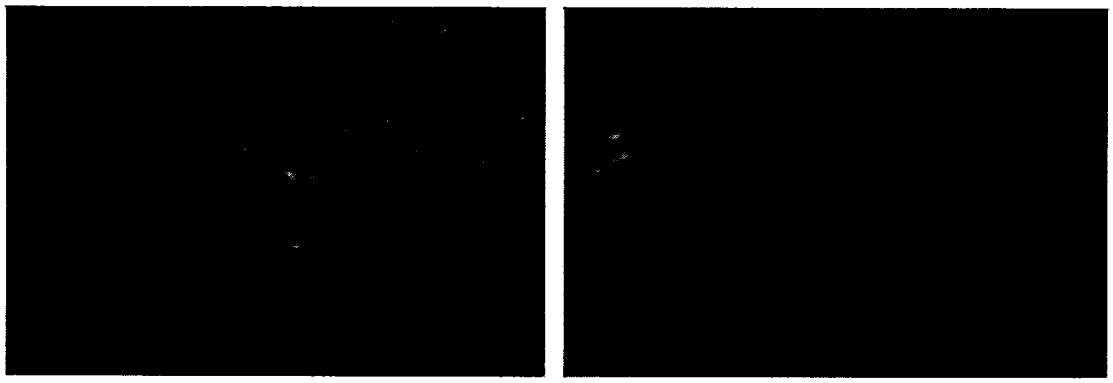


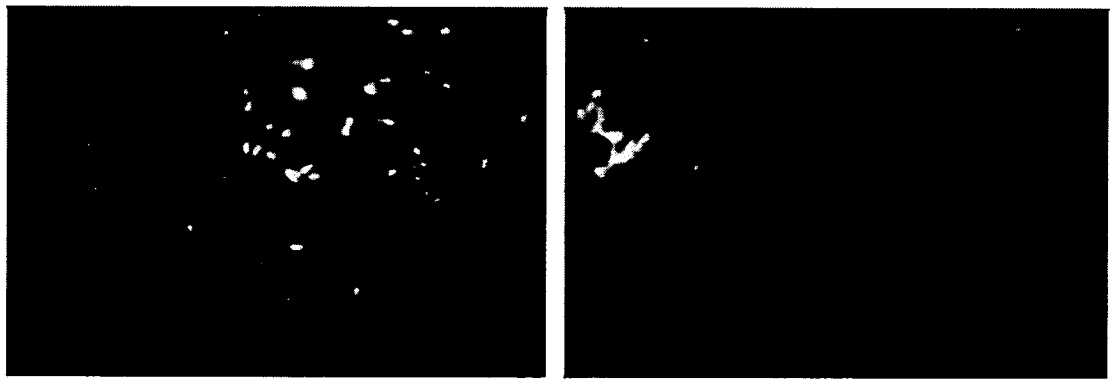
Figure 3. 6 Histogram thresholding applied on the images in Figure 3.5b.

Now let us combine the four pre-processing methods and apply them sequentially on the test images. Figure 3.7 shows the results of the whole pre-processing. Segmentation is performed on the results shown in Figure 3.7d. Note that compared to the original image, the meaningful regions of tissues are better separated and more uniform, which is beneficial to a segmentation algorithm.

(a)



(b)



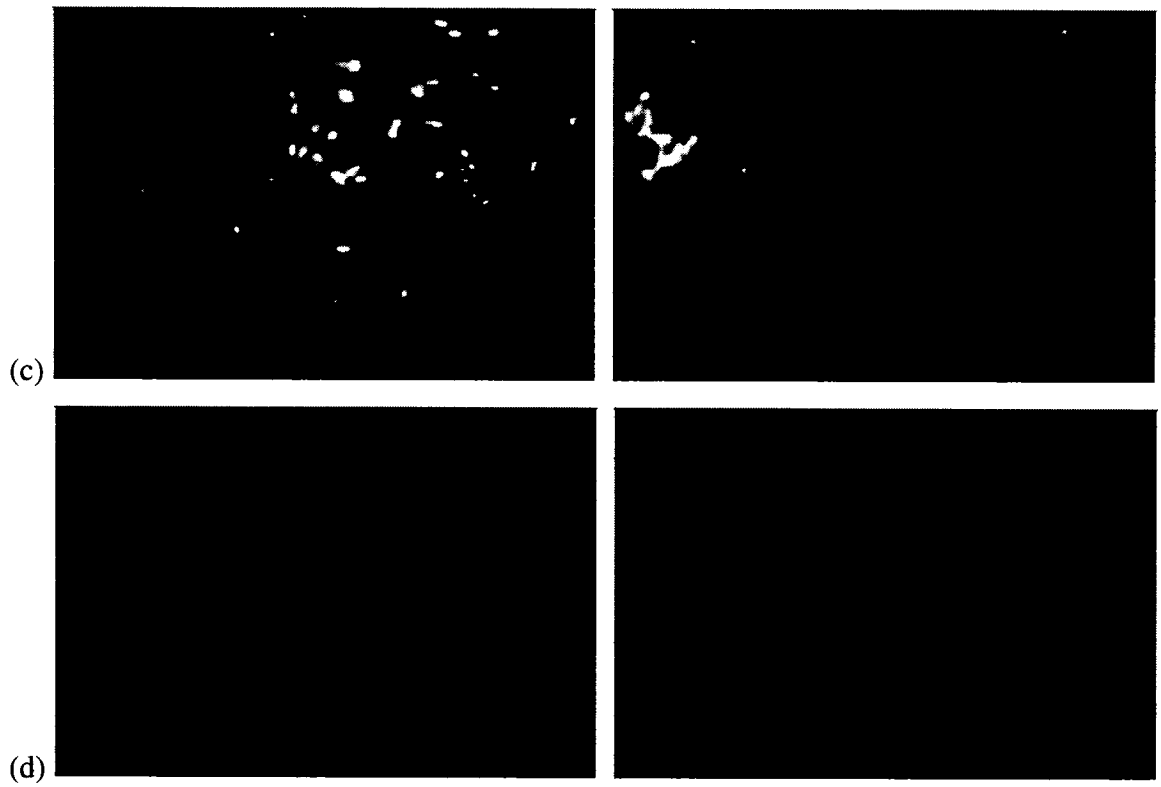


Figure 3. 7 Four pre-processing methods sequentially applied on the images in Figure 3.2. (a) after Gaussian smoothing , (b)after brightness and contrast enhancement, (c) after colour to intensity image conversion, (d) after histogram thresholding .

3.2 Validation of the segmentation algorithm

The proposed segmentation algorithm uses the combined pre-processing results as input, so the same video files were used in testing, and the experimental results obtained from the endoscopic surgical video file are reported here. The parameters have been established experimentally, and some parameters yielding good results for a number of frames are kept to apply to other frames. Results are evaluated and compared qualitatively based on opinions and ground-truth segmentation of a surgeon.

3.2.1 Segmentation experiment results

The efficient graph-based algorithm has only one parameter K. Recall K divided by region size is a factor of minimum internal variation M_{Int}, which decides if two regions

merge or not. So a larger K causes a preference for larger regions. However, K is not a minimum region size. Smaller regions are allowed when there is a sufficiently large difference between neighbouring regions. In multistage region merging algorithm, stage one requires two parameters: a threshold for merging score and minimum common edge length as a portion of the perimeter of the smaller region; stage two and stage three also require two parameters: a threshold for merging score and maximum intensity difference.

Before comparing the results of graph-based segmentation and multistage region merging, we would like to show some results of direct graph-based segmentation (i.e., without pre-processing) and those with the pre-processing described in section 3.1. From Figure 3.8 (direct segmentation) and Figure 3.9 (segmentation after pre-processing), we can clearly conclude to the importance of pre-processing. Without pre-processing, the image is over-segmented as the tissues are not uniform and specular reflections give rise to unwanted regions. In addition to over-segmentation, some more regions are improperly segmented (e.g., segmented regions containing part of tissue, cavity and instrument) because of the low contrast of the image. Note the two figures are obtained using the same parameter K during segmentation, K=100.

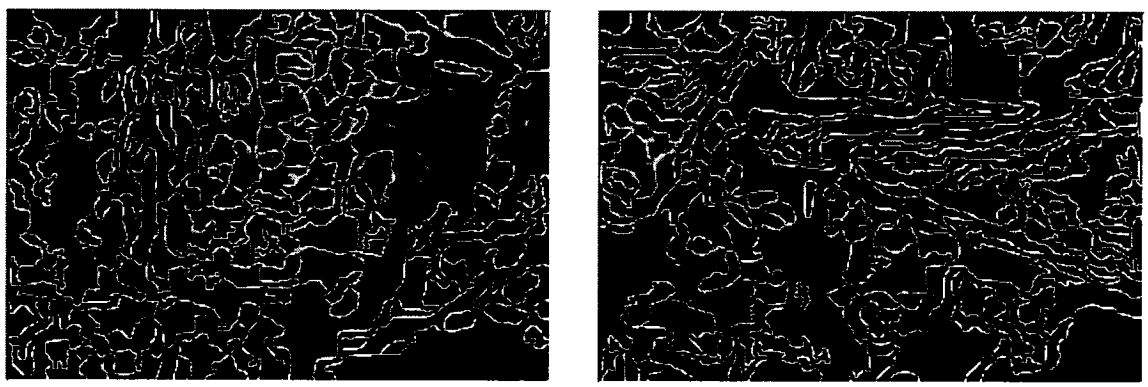


Figure 3. 8 Direct segmentation with the images in Figure 3.2.

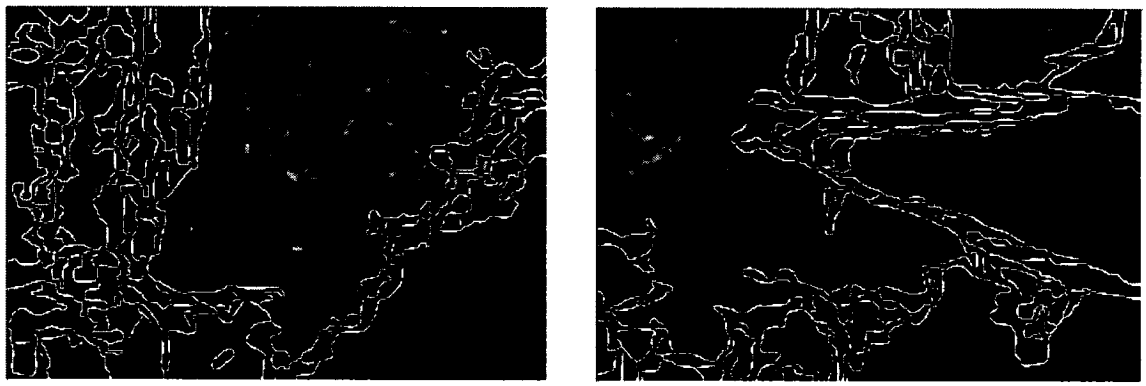
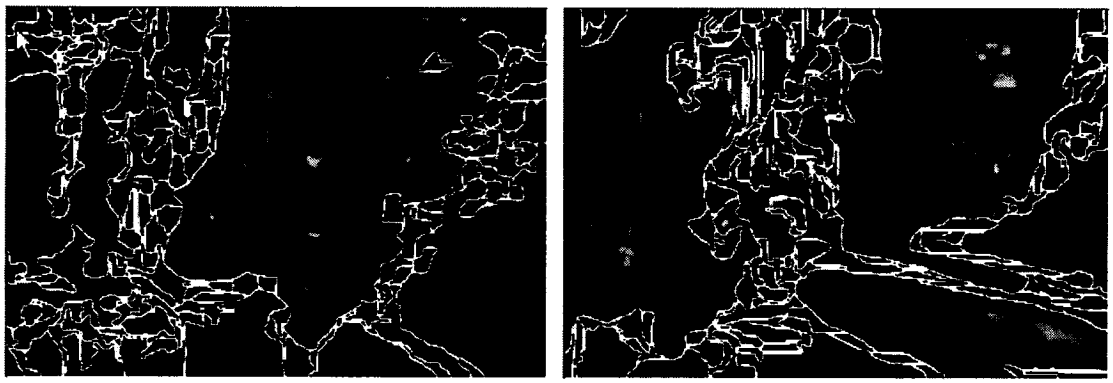


Figure 3. 9 Segmentation after pre-processing with the images in Figure 3.2.

Sample results of endoscopic image segmentations are shown in Figure 3.10, where results of graph-based segmentation are shown from coarse to fine (a-c), and the result with our region merging algorithm (d) following the coarser segmentation in (a). The ground-truth segmentation of the cavity (e) was done manually by a surgeon. Note the multistage region merging is based on the coarse segmentation result but not the fine one because a fine graph-based segmentation generates under-segmented result and the regions are improperly segmented so as to make the result unable to be improved by a region merging algorithm.

(a) 

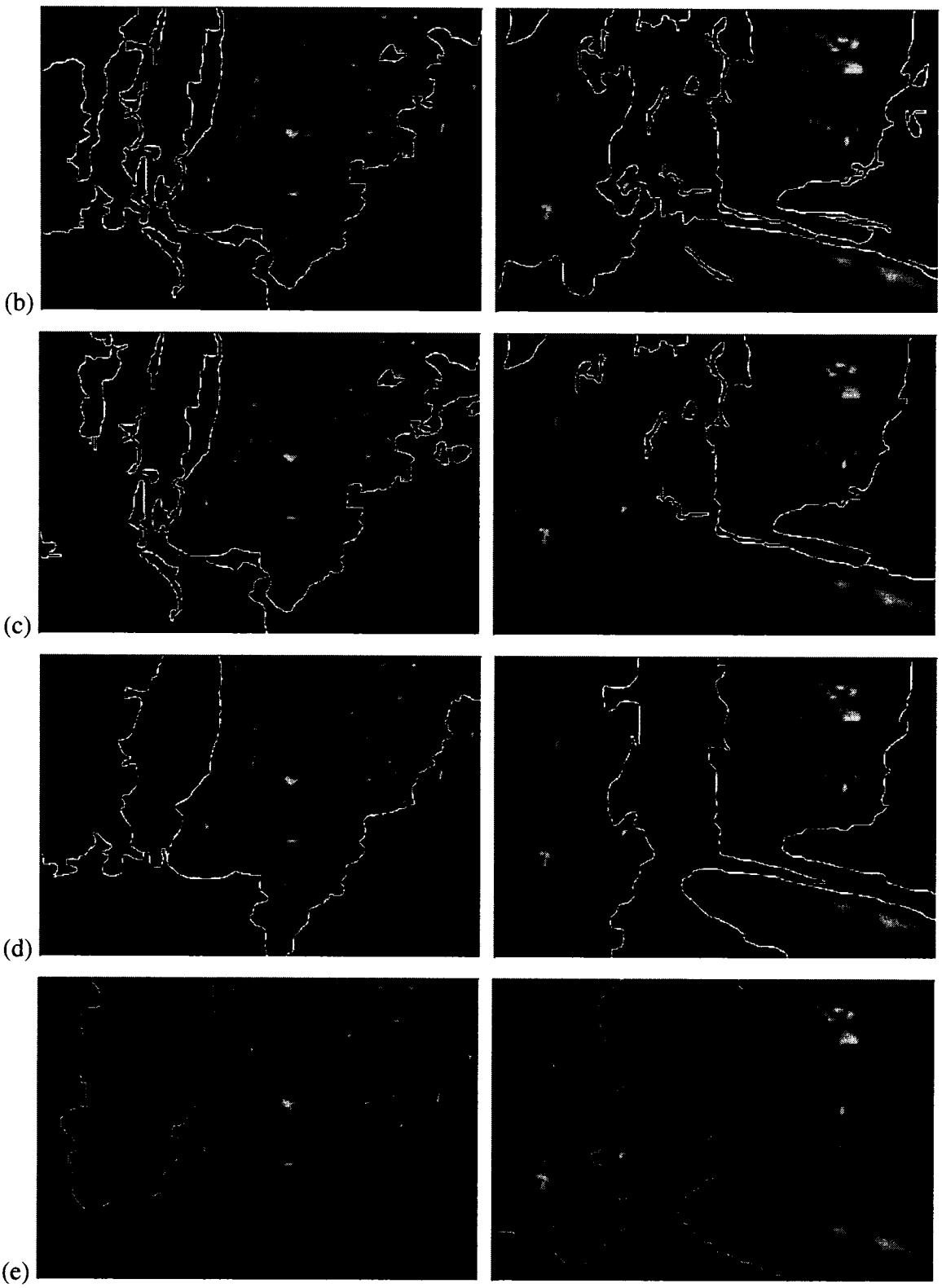

(b)
(c)
(d)
(e)

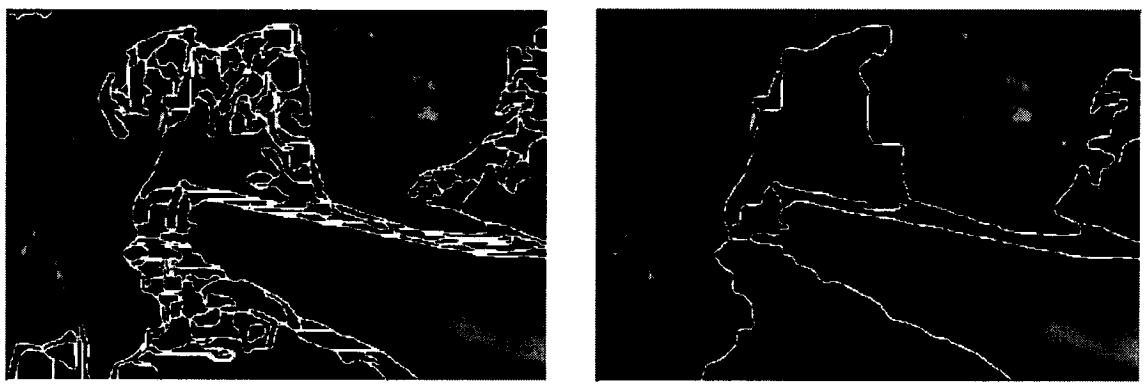
Figure 3. 10 An endoscopic image without instruments (left, frame 1) and another one with instruments (right, frame 534). Region contours are superimposed on original image. (a) Result of coarse graph-based segmentation, K=100. (b) Result of fine graph-based segmentation, K=8000. (c) Result of finer graph-based segmentation, K=12000. (d) Result of our region merging algorithm following the coarse segmentation in (a). (e) ground-truth segmentation of the cavity in the two frames.

Analysing results from Figure 3.10 shows that a finer graph-based segmentation could not improve the quality of the results as the tissues are under-segmented. However, our multistage region merging algorithm helped to partition the image into more meaningful structures or objects (block of tissues, cavity, and instrument). In fact, because of the single criterion used during the whole segmentation, graph-based segmentation could not remove spurious areas inside an object; or it would segment the whole image into few very large regions that are not corresponding to meaningful objects (see Figure 3.10c). On the contrary, our algorithm tries to merge every small region into its neighbours and finally segments the image into several subjectively meaningful large regions. Our experiences show that it is not influenced by small variances in the regions (see Figure 3.10d).

Compared to the ground-truth segmentation (Figure 3.10e), our final segmentation results still have some parts missing. In the left frame, the missing part includes a part of spinal column with very small space to separate it from surroundings. Actually it looks more homogeneous as the tissues to its left. Obviously it is hardly achievable for an unsupervised segmentation algorithm based on general features (e.g., intensity) and considering a single frame to segment the spinal column as part of the cavity. In the right frame, small parts at the left and at the right are missing. That is understandable since they are rejected because of intensity difference and intersection length. Some missing parts could be recovered if we optimize merging parameters for the particular frames, because some of them are individual regions in the source segmentation map (Figure 3.10a); but the parts that have already been segmented into

other big regions are not recoverable, like the spinal column in the left frame and the right part in the right frame. In fact, the ground-truth segmentation (Figure 3.10e) is not as clear as the name implied, because the cavity area is not a real object, and the surgeon has had difficulty to identify it by just looking at one frame – he had to look at a part of the video sequence to find out the operational area of concerned and specify it as the cavity. So what is expected for the cavity is very difficult to obtain, at least at a frame by frame basis. We consider that what we have obtained is a good approximation, and an important first step for the exact segmentation of the cavity, as the result does correspond to the main part of the cavity.

Figure 3.11 shows another set of images including intermediate results of multistage region merging. Note that merging stage one did most of the work of forming homogeneous regions (Figure 3.11b), but there are still some small regions left because the compound score of similarity and intersection is within the threshold. Then at merging stage two (Figure 3.11c), small regions are merged into large neighbours as long as their intensity differences are not too big. At the last stage (Figure 3.11d), further merging happens when a region is almost contained in a neighbour region or it is adjacent to a relatively very large neighbour. In this case, the final result is very close to the ground-truth segmentation (Figure 3.11e), in which the cavity is split into two parts by the instrument, corresponding to two regions in the merging result (Figure 3.11d).



(a) (b)

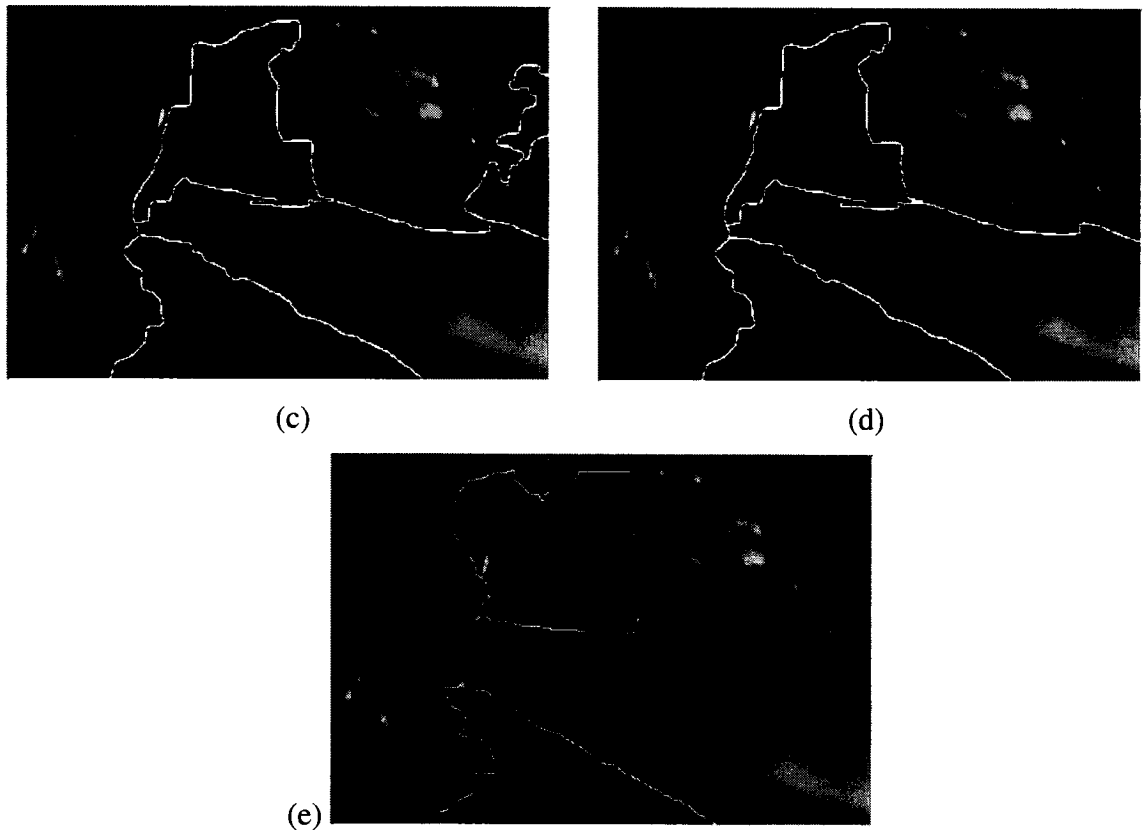


Figure 3. 11 Results of multistage region merging at frame 463. (a) Result of coarse graph-based segmentation, K=100. (b) Result of merging stage one. (c) Result of merging stage two. (d) Result of merging stage three – final result. (e) ground-truth segmentation of the cavity.

A fine graph-based segmentation cannot obtain the same result as our merging algorithm – this is demonstrated in Figure 3.12 with four finer results from graph-based segmentation on the same image frame as in Figure 3.11. Note the graph-based segmentation uses one intensity criterion throughout the process and merging happens between two least intensity difference regions, regardless their size and intersection. As a consequence, very small regions still exist even when almost all other regions have been merged into one region (see Figure 3.12d) just because of big intensity difference. We can also notice the different region forming with different K, which reflects the tolerance

to intensity difference in one region. As it can be observed from this result, our algorithm does handle the drawbacks of graph-based segmentation.

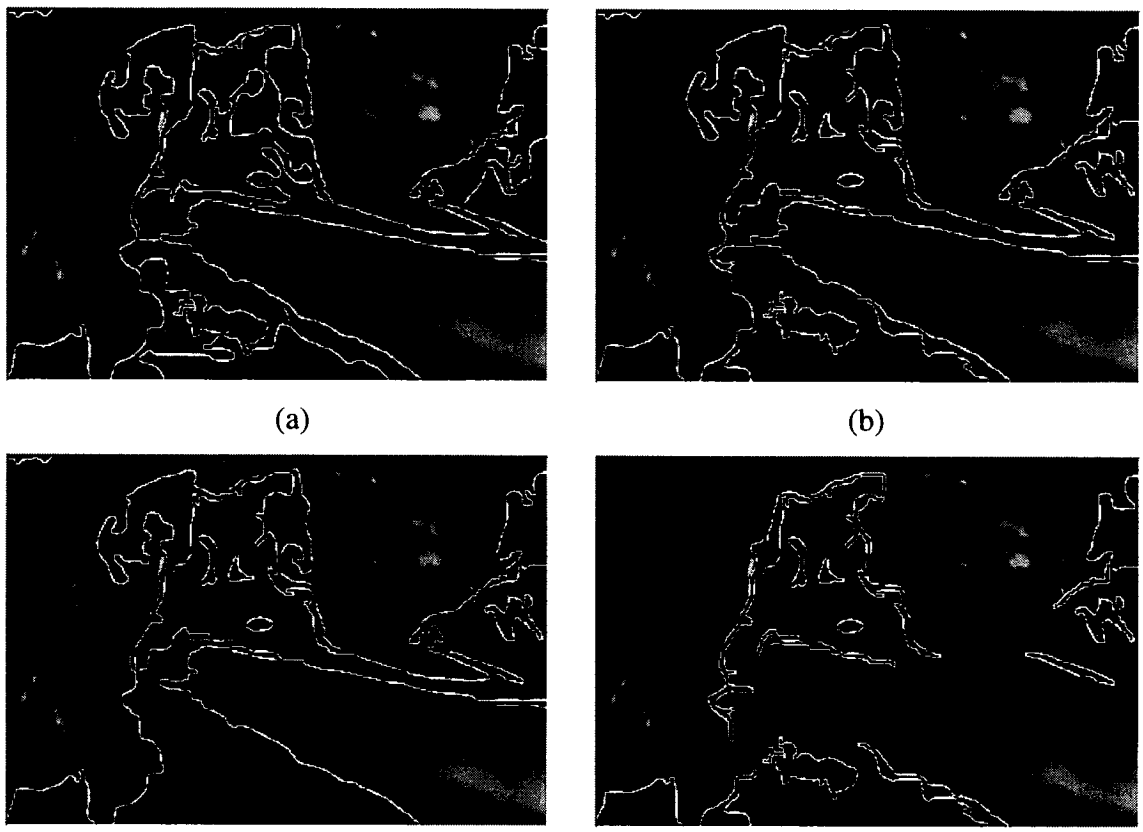
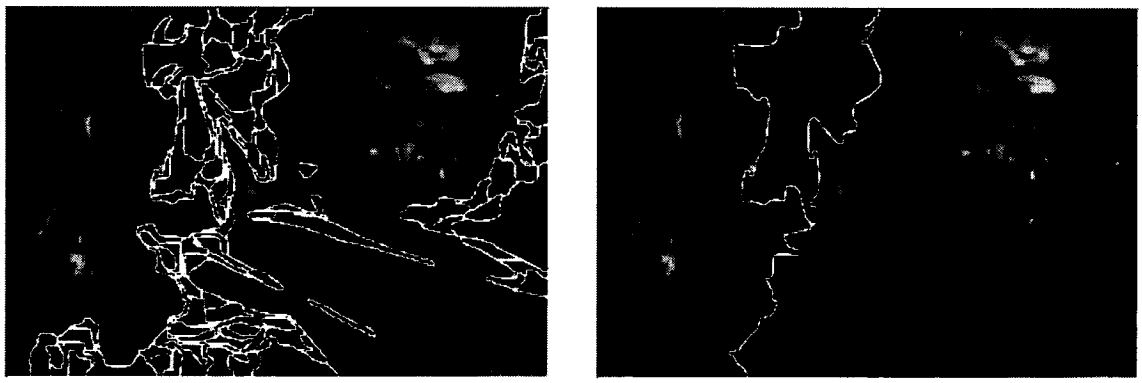


Figure 3. 12 Results of graph-based segmentation at frame 463. (a) K=1000, (b) K=5000, (c) K=10000, (d) K=50000.

Our merging algorithm is not sensitive to the initial segmentation as long as it is not too fine (under-segmented). Hence, the parameter K for the graph-based segmentation should be moderate to result in a coarse segmentation. We obtained good final segmentation results by setting K = 300 for many image frames. The parameters for region merging are more complicated, we need to decide the thresholds and other parameters by prior knowledge and experiment. In fact, parameters need to be fine-tuned according to the image characteristics, but once they are set, they can be applied to similar images (in our case, complete sequence of endoscopic images). During the

experiments, we obtained the same final result with a large range of K (K=100, 500 or 1000) and fixed region merging parameters.

Although multistage region merging can significantly improve the segmentation results, it does not guarantee success all the time. Note that the merging is based on the output of initial segmentation, so the final result may not be perfect if the initial segmentation has under-segmented some regions. This happens especially on the endoscopic surgical images with instruments in the field of view (Figure 3.13), because the reflection on the instrument makes its colour or intensity very close to the surrounding tissues. In this case we consider using some other ways to refine the boundary, such as the contour modification method used in [67]; or using a special filter to make the difference between the instruments and surrounding tissues large enough for a correct segmentation.



(a) (b)

Figure 3. 13 False segmentation of an endoscopic image with instruments. (a) Result of coarse graph-based segmentation, K=100. (b) Result of multistage region merging following the coarse segmentation.

3.3 Validation of hybrid snakes tracking algorithm

Cavity tracking follows segmentation and the hybrid snakes tracking algorithm uses the contour of the cavity area identified in the segmentation map as initial contour; it also uses the combined pre-processing steps to process current frame, and the contour inherited from previous frame is deformed based on the processed image. The parameters have been established experimentally and used throughout the tracking process. Results

are evaluated qualitatively based on the initial segmentation result and changing of the cavity boundary in the video sequence.

3.3.1 Experimental results

The tracking tools are mainly composed of traditional snake and GVF snake. They are based on the function cvSnakeImage in OpenCV. The function updates snake in order to minimize its total energy that is a sum of internal energy depending on contour shape (the smoother contour is, the smaller internal energy is) and external energy depending on the energy field, which reaches minimum at the local energy extremums that correspond to the image edges in case of image gradient. For each contour point, the function searches in a neighbourhood to find a point that minimize the energy of whole contour points, and then replace the point with the new one (or move the point to the new position). It works iteratively until convergence (no more points moved) or maximum iterations reached.

Parameters required by snake function include contour points, weight of continuity energy (α), weight of curvature energy (β), weight of image energy (γ), size of neighbourhood and termination criteria. Besides, the image energy field (external force) is also an input. When it is an intensity gradient field, the function works as a traditional snake. When it is a GVF field, the function works as a GVF snake. The initial contour is the contour of a region in the final segmentation map, which can be simply identified by a click inside the region. Actually, the region boundary obtained from our dedicated segmentation algorithm is already very close to the real boundary in many situations. The snake function outputs a new contour composed of the moved points, which may not be aligned with the raster line. To best reflect the unstructured shape of the cavity area, we would like to use as many points as possible on the contour during snake deformation. So an interpolation algorithm is applied on the deformed contour to remove duplicated points and to insert new points between two non-connected neighbouring points. The intermediate points are found using 8-connected Bresenham algorithm.

As mentioned in section 1.5.5 the image gradient based external force dies out quickly away from the center of edges, so traditional snake has a very small capture range and it requires the initial contour to be close to the object boundary we want to detect.

Gradient vector flow is also based on the gradient, but it diffuses it to make the force gradually fall off away from the center of edges, so GVF snake has a larger capture range. Figure 3.13 shows the computed gradient field and GVF field of the images in Figure 3.2. A brighter pixel indicates a stronger force in the field. We can see the different distribution of forces in Figure 3.14. It is also noticeable that the gradient field includes more details than the GVF filed. This is not what is needed for snake deformation in our application, because in area with many details, there are many different forces, which may give false guidance to a contour pass through the area. In fact, in our cavity tracking, traditional snake deforms the initial contour unexpectedly. The contour breaks out of the cavity area in several frames. So traditional snake loses tracking quickly, but GVF snake works better because the contour could stay in the cavity area for many frames.

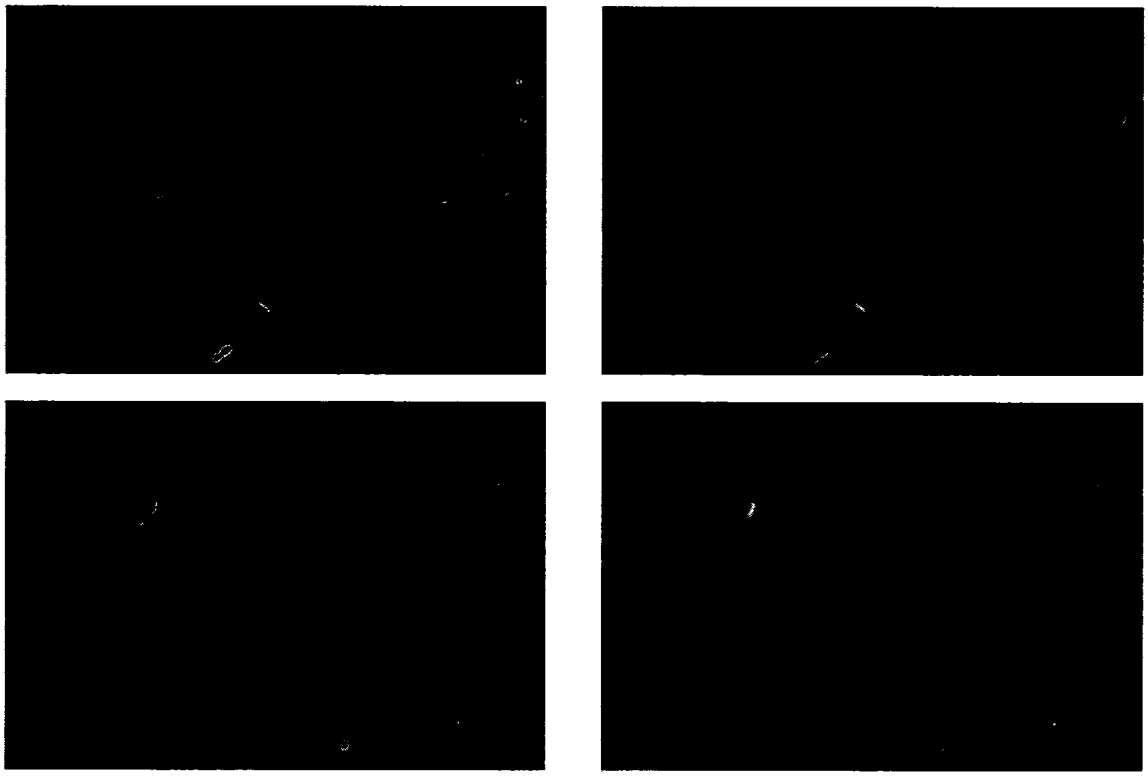
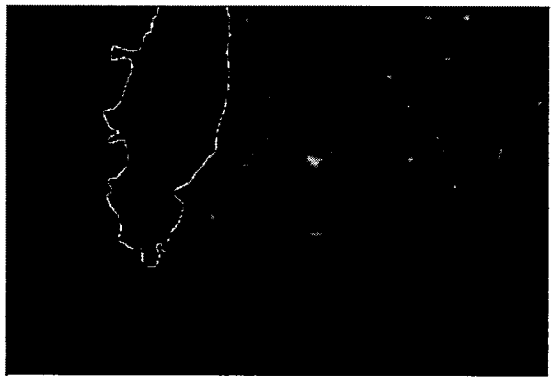


Figure 3. 14 Gradient field and GVF field of the images in Figure 3.2. left: gradient field, right: GVF field.

If GVF snake performs better than traditional snake, why we still use traditional snake in our application? Recall that GVF filed has the ability to guide the snake moving into concave areas, but we do not really want our object (the cavity area) to extrude into concave regions of neighbouring structures. Traditional snake is not sensitive to boundary concavities, and its internal forces can help to smooth the contour. So we can make use of traditional snake to regulate the contour by giving larger weight to the internal force, then use GVF snake with larger weight on external force to further move the contour to strong edges in a larger range. In a word, utilization of traditional snake with appropriate parameters can smooth the contour, and prevent GVF snake moving into concavities, which is what we need for cavity tracking. Figure 3.15 shows the effect of regulation by traditional snake, and gives the result of GVF snake deformation from the original contour and from the regulated contour.



(a)

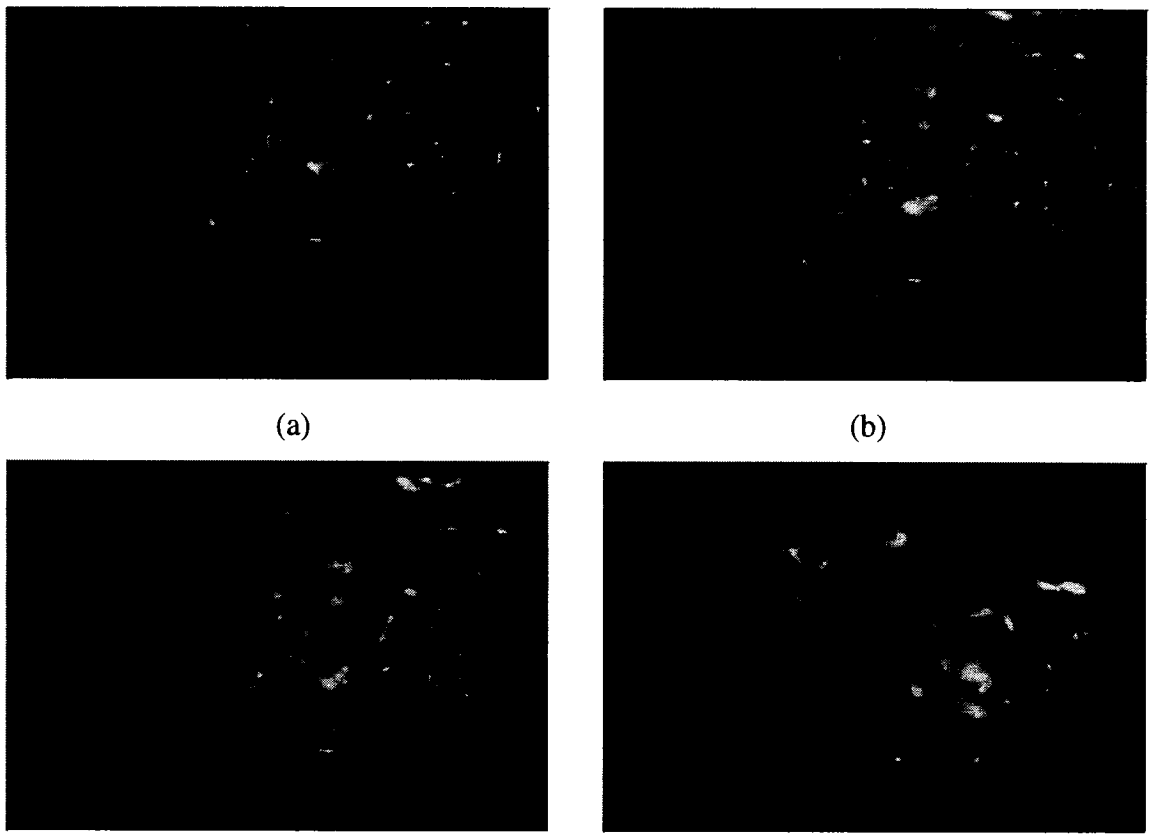
(b)

(c)

(d)

Figure 3. 15 Effect of contour regulation by traditional snake. (a) original contour, (b) regulated contour by traditional snake, (c) GVF snake deformation from the original contour, (d) GVF snake deformation from the regulated contour.

Using a combination of traditional snake and GVF snake, we have tracked the cavity area in an endoscopic video sequence for 64 frames (the best case). Actually the snake continues tracking until an abrupt change happens (an instrument moves into the sight and occludes a large part of the cavity area). The parameters used in the traditional snake are as follows: α=3.5, β=1.5, γ=0.5, neighbourhood size is 5×5, maximum iterations (serve as termination condition) is 15. For GVF snake, α and β are the same, γ=6.5, neighbourhood size is 15×15. So we give relatively larger weight to internal force (α, β) in the traditional snake to allow it to regulate the contour in a small neighbourhood, and give relatively larger weight to external force (γ) in the GVF snake with larger neighbourhood size to allow it to deform the contour in a larger range. Figure 3.16 shows four sample frames tracked with the combined snakes. The above mentioned 64 consecutive frames are the best situation for tracking, because the entire cavity presents in these frames. The cavity could be as complex as including several different regions, and the change from frame to frame could be large too. So the combined snakes could lose tracking in two frames in the worst case, such as from frame 64 to frame 65. Using GVF snake alone, it can deform the contour to fit current frame as the combined snake does, but the resulting contours are more irregular and some may include long extrusion parts. Using traditional snake alone, with the same parameters as GVF snake, it just deforms the contour slightly each time, so the contour has nearly no change from frame to frame. In fact, we found the traditional snake could not deform the contour to fit current frame effectively, even in the best situation, i.e., the 64 frames with full cavity, part of the deformed contour falls outside of the cavity area just in 3 frames.



(a) (b)

(c) (d)

Figure 3. 16 Tracking result by combined snakes. (a) frame 1, (b) frame 16, (c) frame 30, (d) frame 48.

The lost tracking criterion used in the tracking example is the average intensity difference between the deformed contour area and the original contour area. We look it as lost tracking if the difference is more than 50% of the original value. Actually the tracking algorithm can be used to track any region, but lost tracking condition may not be the same. So we designed a parameter adjustment dialog box to allow easy change of parameters such as intensity difference for lost tracking, Figure 3.17 is a screen shot of the dialog box. We also set an option to enable/disable traditional snake in the program, because sometimes it is not necessary to run the traditional snake to smooth the contour. Thus the application allows us to adjust parameters or even to reconfigure functional modules on the fly.

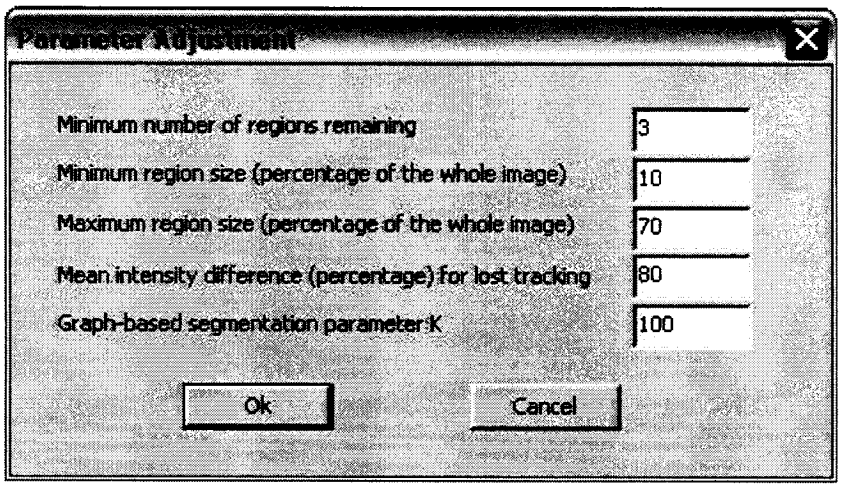


Figure 3. 17 Parameter adjustment dialog box.

One thing about the tracking that is not satisfying and needs improvement is that the deformed contour does not fit the true boundary well. That is mainly because the GVF snake we implemented does not perform as expected. The original GVF snake was implemented in Matlab by the authors [71]. It uses matrix operations to calculate the energy function with x and y component of GVF field, and moves contour points until convergence (a stable status is reached). Here we compute the magnitude of GVF field as external force in the greedy algorithm [72] implemented in OpenCV. The function cvSnakeImage can assign appropriate positions for the contour points after deformation if you use the default external force, i.e., the gradient field computed inside the function; but it generates many duplicated points if you use another external force. In our experiment, an input contour including hundreds of points turns to be tens of points after the GVF snake deformation (the other points are still there, but are duplicates). The larger weight you put on the external force, the fewer useful points you get (so we cannot put any weight as we want). That might be a bug in OpenCV, or we misused the function. Anyway, those sparse points cannot form an entire contour, so we have to interpolate points to fill the gaps; but the interpolated contour cannot be a good representation of the real contour if the points are too sparse. The errors will accumulate when the interpolated contour serves as the initial contour of next tracking cycle. Actually the GVF field is

quite good as we have seen in Figure 3.14, and it should be able to guide the snake to a place very close to the real boundary. We believe the result would be better if a C version GVF snake being implemented as the author proposed in [71]. This will be the subject of future works.

3.4 Complexity of the algorithms and computation time

Each of the four pre-processing steps computes a new value for each image points, so the complexity of pre-processing is $O(n)$, where n is the number of pixels in an image frame. The complexity of graph-based segmentation is $O(n \log n)$, as concluded in [5]. Each stage of the multistage merging algorithm is similar to the RSST algorithm, and has a complexity of $O(k_1 m^2)$, where m is the number of regions in the input segmentation map (much less than n), and k_1 is a factor. We add the factor k_1 to highlight that our region merging algorithm is more complicated than RSST, as we have to search the segmentation map (same size as the original image) to find regions neighbouring to the region under consideration and compute their properties; we also need to update the segmentation map when a merging happens. The complexity of snake algorithm is $O(k_2 n)$, where k_2 is a factor related to termination condition and window size. GVF snake is slower than traditional snake as the calculation of GVF field is more complex than gradient field. Based on our tests on a Pentium IV 2.4GHz PC with 352 x 240 endoscopic images, the typical running time is around 0.01s for the combined pre-processing, 0.2s to 0.3s for the graph-based segmentation, 6s to 30s for the multistage region merging, and 0.8s to 1.3s for the hybrid snakes tracking. The multistage region merging takes the longest time, in which the number of regions participated in the merging plays an important role. Table 3.1 gives some running time samples of each merging stage and the number of regions.

Table 3. 1 Sample running time of multistage region merging

Frame #	Merging stage	Number of regions	Running time (seconds)
15	Stage 1	124	23.6
	Stage 2	19	1.14
	Stage 3	8	0.56
151	Stage 1	139	15.07
	Stage 2	6	0.05
	Stage 3	5	0.04
312	Stage 1	96	9.27
	Stage 2	14	0.6
	Stage 3	5	0.08
505	Stage 1	58	5.95
	Stage 2	19	1.0
	Stage 3	7	0.19
677	Stage 1	84	7.17
	Stage 2	9	0.28
	Stage 3	4	0.03
827	Stage 1	67	12.99
	Stage 2	25	1.18
	Stage 3	5	0.15

Chapter 4 Conclusion and Future Work

4.1 Conclusion

In this thesis, a system for tracking the cavity area in endoscopic video sequences has been presented. The primary objective of the proposed scheme is segmentation, identification and temporal tracking of the inter-vertebral disc of the spine appearing as a cavity in thoracoscopic images. We implemented the system that can be used in an unsupervised fashion, in which only simple user intervention is required to accomplish the tracking task.

Endoscopic images obtained in the context of thoracic discectomy are very challenging. To fulfill the objectives, we divided our work into two subtasks: (1) finding the precise boundary of the cavity in a frame, and (2) tracking the cavity boundary from frame to frame. The first subtask was accomplished by means of image pre-processing and graph-based segmentation followed by multistage region merging. The second subtask was accomplished by use of a combination of traditional snake and GVF snake. The main contribution in this work is the special designed multistage region merging algorithm that can overcome usual graph-based segmentation drawbacks and help to segment the entire cavity area required for tracking, as well as the combined snakes that can keep a large capture range without suffering from spreading noises during tracking process.

We have introduced a dedicated segmentation method combining graph-based segmentation and multistage region merging. The method first uses an efficient variant of the RSST algorithm to do an initial coarse segmentation, and then uses an original three-step method to further merge regions into more meaningful structures or objects. Experimental results on endoscopic images of a thoracic discectomy procedure show that our method can segment the cavity adequately for a variety of challenging images. Compared to ground truth segmentation as indicated by a surgeon, some parts of the cavity are sometimes missing. However, the surgeon has had difficulty to identify the cavity correctly in still images. He had to rely on temporal information to identify the

cavity. Hence, in future work, temporal information should be used more specifically to this aim. Nevertheless, in contrast to graph-based segmentation method using a single criterion, our multi-criteria merging method shows significant improvements on final results. We have also shown how at each stage the segmentation is improved by removing unwanted regions.

After running the specially designed segmentation method on a frame, the cavity area is well separated from other regions, although some parts might be missing. It can then be identified by a user to initiate tracking. Only a mouse click inside the cavity area on the segmentation map will give enough information for the system to find out the boundary that is ready for tracking.

Tracking of the boundary identified in last step is accomplished by snake method. Snake model has the ability to deform an initial contour obtained in a frame and to fit it in neighbouring frames of a video sequence, which makes it a popular tool for boundary tracking. It uses internal forces to smooth the contour and external forces to move the contour to desired image features such as edges. We have implemented a combination of traditional snake and GVF snake to track the cavity. Traditional snake uses intensity gradient field as external force, and it has a small capture range. GVF snake uses GVF field as external force, and it has a larger capture range; but tends to amplify the extruding parts on the contour, which is not desirable for cavity tracking. Our tracking strategy is to run traditional snake before running GVF snake. Relatively larger weight is given to internal force in the traditional snake to allow it to regulate the contour, and relatively larger weight is given to external force in the GVF snake with larger neighbourhood size to allow it to deform the contour in a larger range. By this way, we are able to keep a large capture range without suffering from spreading noises. Experimental results show that the combined traditional snake and GVF snake can effectively track the cavity in a number of frames. We have also designed a lost tracking logic to allow an easy restarting of the segmentation and tracking process in case of the tracking lost due to scene cut or other special events.

4.2 Future work

The work presented in this thesis can be improved and extended in various ways. Here is a non-exhaust list of possible improvements.

- Improve the quality and robustness of segmentation

  As mentioned in section 3.2, sometimes the segmentation results are not satisfactory, particularly when operating instruments are in the field of view of the endoscope. That is mainly because the reflection on the instrument makes its colour or intensity very close to the surrounding tissues. The reflection and other factors can mix up region boundaries, which results in a false initial segmentation by the graph-based method. To improve the initial segmentation result, we may change the pre-processing method, e.g., using a special filter to make the difference between the instruments and surrounding tissues large enough for a correct segmentation. Adjusting parameter K is also useful in some cases. If the final segmentation result does not fit the region boundaries well, we may need to adjust the parameters for multistage region merging, or try some other ways to refine the boundary, such as the contour modification method proposed in [67].

- Improve segmentation by validating with the correlations between frames

  There exist correlations between frames other than motion information, such as the relative position of structures. Recall the surgeon needed to identify the cavity by looking at the sequence. He was probably looking for some special features (e.g. shape) and correlations between frames (relative position of structures). The segmentation results may be improved if this information can be used.

- Improve the efficiency of multistage region merging

  Although we have adopted an efficient graph-based method for segmentation, the multistage region merging process is relatively slow. As an example, when the segmentation is done in a second, the region merging takes almost one minute to give the final result. The efficiency of the algorithm has to be improved before it can be used in real-time applications. Let us analyse the algorithm to see the possible ways to improve its efficiency. The region merging algorithm finds the best merging target

iteratively in each of the three stages. That is similar to RSST algorithm. At each iteration, it first finds the least size region and all its neighbouring regions. Next, it calculates those regions boundary length (or perimeter) and the length of common edges between the least size region and one of its neighbouring regions. After all of the region data needed for calculation of merging score is ready, it then calculates merging scores and tries to merge two regions at a time. It also needs to update the segmentation map if a merging happened. So the merging process merges regions from small to large recursively, and it is a computation-intensive process. Reusing of data is a common strategy to improve the efficiency of this kind of process. The region data, such as size, weight and standard deviation, are used repeatedly in the calculation of merging score. We have already optimized the code to avoid unnecessary computation of those region data, but the data related to two regions like common edge length is still computed dynamically because the regions change during merging. Currently finding neighbouring regions of a region is also done dynamically because of the same reason. There may be a way to reuse the data related to two regions, but it seems that is not an easy task. Another direction to optimize the algorithm is to reduce the number of iterations. That could be done by removing some loops or reducing the number of regions participated in comparison and calculation, as long as the quality of the final result is not affected significantly.

- Eliminate the need of user interaction

  The identification of cavity area is done by user interaction in our application. The ideal situation is to identify the cavity and to track it automatically. The unsupervised identification could be achievable by making use of some statistical data, such as average intensity, location, size, shape, etc. Another situation needs user intervention in our application happens when the tracking of original contour is lost. Currently we use a simple average intensity criterion to evaluate if the tracking loses or not. That is not accurate actually; we may improve the lost tracking criterion by incorporating statistical data of the tracked object as well. Furthermore, we could even eliminate the

user intervention on lost tracking if the criterion to identify the tracked object (the cavity in our case) is reliable.

- Improve the quality and robustness of tracking

  We mentioned in section 3.3 that the GVF snake we implemented does not work as desired. A lot of contour points overlap after each deformation by the GVF snake. That situation would impair the performance of the snake, though we try to interpolate the deformed contour to mend it. We may investigate the snake implementation in OpenCV library to find out why that happens, but possibly the better way is to implement the C version of GVF snake in our application according to the original version designed in Matlab.

References

[1] D. R. Uecker, C. Lee, Y. F. Wang and Yulun Wang, "Automated Instrument Tracking in Robotically-Assisted Endoscopic Surgery," Journal of Image Guided Surgery, Vol. 1, No. 6, 1996.

[2] Y. F. Wang, D. R. Uecker and Yulun Wang, "A New Framework for Vision-Enabled and Robotically-Assisted Minimally Invasive Surgery," Computerized Medical Imaging and Graphics, vol. 22, no. 6, pp. 429-437, 1998.

[3] M. Kass, A. Witkin and D. Terzopoulos, "Snakes: Active contour models," Int. J. Comput. Vision, 1(4), 321–331, 1988.

[4] T. McInerney and D. Terzopoulos, "Deformable models in medical image analysis: a survey," Medical Image Analysis, vol.1, No. 2, pp. 91-108, 1996.

[5] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient Graph-Based Image Segmentation", *IJCV(59)*, No. 2, pp. 167-181, September 2004.

[6] C. Xu and J. L. Prince, "Snakes, shapes, and gradient vector flow," IEEE Trans. on Image Processing, vol. 7, pp. 359-369, Mar. 1998.

[7] R. M. Haralick and L. G. Shapiro, Computer and Robot Vision, Addison-Welsey, New York, 1992.

[8] M. Sonka, V. Hlavac and R. Boyle, Image Processing, Analysis and Machine Vision, 2nd edition, PWS Boston, 1998.

[9] W. K. Pratt, Digital Image Processing, John Wiley & Sons, 1991.

[10] T. Y. Young and K.-S. Fu, Handbook of Pattern Recognition and Image Processing, Academic Press, 1986.

[11] W. E. Snyder and H. Qi, Machine Vision, Cambridge University Press, 2004.

[12] J.S. Weszka, "A survey of threshold segmentation techniques," Computer Graphics and Image Processing, Vol. 7, pp. 259-265, 1978.

[13] G. Bilbro and W. Snyder, "Optimization of Functions with Many Minima," IEEE Trans. on Systems, Man and Cybernetics, 21(4), July/August 1991.

[14] L. Li, J. Gong and W. Chen, "Gray-level image thresholding based on Fisher linear projection of two-dimensional histogram," Pattern Recognition, vol. 30, no. 5, pp. 743-749, 1997.

[15] J. M. Prager, "Extracting and labeling boundary segments in natural scenes," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 2, no. 1, pp. 16-27, 1980.

[16] J. Canny, "A Computational Approach to Edge Detection," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 8, No. 6, Nov 1986.

[17] J. Bryant, "On the clustering of multidimensional pictorial data," Pattern recognition, Vol. 11, pp. 115-125, 1979.

[18] T. Vlachos and A.G. Constantinides, "A graph-theoretic approach to colour image segmentation and contour classification," Proc. Int. Conf. Image Processing and its Applications, pp. 298-302, 1992.

[19] M. Suk and S. M. Chung, "A new image segmentation technique based on partition test," Patter Recognition, Vol. 16, No. 5, pp. 469-480, 1983.

[20] H. Frigui and R. Krishnapuram, "A robust competitive clustering algorithm with applications in computer vision," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 21, no. 5, pp. 450-465, 1999.

[21] J. Serra, "Image Analysis and Mathematical Morphology," Academic Press, 1982.

[22] F. Moscheni, "Spatio-Temporal Segmentation and Object Tracking: an Application to Second Generation Video Coding," PhD thesis, Swiss Federal Institute of Technology, Lausanne, Switzerland, 1997.

[23] B. D. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," DARPA Image Understanding Workshop, 1981.

[24] F. Meyer and S. Beucher. "Morphological segmentation," Journal of visual communication image representation, Vol. 1, No. 1, pp. 21-46, September 1990.

[25] Y. Tsaig and A. Averbuch, "Automatic Segmentation of Moving Objects in Video Sequences: A Region Labeling Approach," IEEE trans. on circuits and systems for video technology, vol. 12, no. 7, July 2002.

[26] S. Y. Chien, Y. W. Huang, S. Y. Ma, and L. G. Chen, "Automatic Video Segmentation for MPEG-4 Using Predictive Watershed," IEEE International Conference on Multimedia and Expo, 2001.

[27] C. Gu, "Multivalued morphology and segmentation-based video coding," PhD thesis, Swiss Federal Institute of Technology, Lausanne, Switzerland, 1996.

[28] R. O. Duda and P. E. Hart, Pattern Classification and Science Analysis, John Wiley & Sons, New York, 1973.

[29] S. Khan and M. Shah, "Object Based Segmentation of Video Using Color, Motion and Spatial Information," IEEE 2001.

[30] Y. Wang, T. Tan and K.-F. Loe, "Video Segmentation Based on Graphical Models," Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003.

[31] D. G. Lowe, Perceptual Organization and Visual Recognition, Kluwer Academic Publishers, Boston, Mass., 1985.

[32] B. P. L. Lo, A. Darzi and G. Yang, "Episode Classification for the Analysis of Tissue/Instrument Interaction with Multiple Visual Cues," MICCAI (1) 2003, pp. 230-237.

[33] J. Boisvert, F. Cheriet and G. Grimard, "Segmentation of Endoscopic Images for Computer Assisted Surgery," SCIA03, pp. 587-594.

[34] R. Castagno, T. Ebrahimi and M. Kunt, "Video Segmentation Based on Multiple Features for Interactive Multimedia Applications," IEEE Trans. On Circuits and Systems for Video Tech., Vol. 8, No. 5, September 1998.

[35] E. Chalom and V. M. Bove, Jr., "Segmentation of Frames in a Video Sequence Using Motion and Other Attributes," SPIE Digital Vodeo compression: Algorithms and Technologies, 2419, pp. 230-241, 1995.

[36] J. I. Khan, Z. Guo and W. Oh, "Motion based object tracking in MPEG-2 video stream for perceptual region discrimination rate transcoding," Proceedings of the 9th ACM international conference on multimedia. pp. 572-576, 2001.

[37] R. C. Jones, D. DeMenthon, D. S. Doermann, "Building mosaics from video using MPEG motion vectors," 1999 ACM.

[38] H. Zhu and Z. Li, "A Video Segmentation Algorithm Based on Spatial-Temporal Information," IEEE 2002.

[39] W. Krattenthaler, K. J. Mayer and M. Zeiller, "Point Correlation: A Reduced-Cost Template Matching Technique," Proc. ICIP, pp. 208–212, 1994.

[40] A. K. Jain, Y. Zhong and S. Lakshmanan, "Object Matching Using Deformable Templates," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 18, no. 3, pp. 267–278, Mar. 1996.

[41] R. Curwen and A. Blake, "Dynamic Contours: Real-Time Active Splines," In Active Vision Ed Blake, Yuille, MIT Press, 1992.

[42] M. P. Dubuisson, S. Lakshmanan and A. K. Jain, "Vehicle Segmentation and Classification using Deformable Templates," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 18, no. 3, pp. 293-308, 1996.

[43] S. Osher and J. A. Sethian, "Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulations", Journal of Computational Physics 79, pp. 12-49, 1988.

[44] J. Suri, L. Liu, S. Singh, S. Laxminarayan, X. Zeng, and L. Reden, "Shape recovery algorithms using levels sets in 2-D/3-D medical imagery: A state-of-the-art review", IEEE Transactions on Information Technology in Biomedicine 6(1), 2002.

[45] G. B. Coleman and H. C. Andrews, "Image segmentation by clustering," Proc. IEEE, 5:773–785, 1979.

[46] S. Nitsuwat, J. S. Jin and H. M. Hudson, "Motion-based video segmentation using fuzzy clustering and classical mixture model," IEEE 2000.

[47] D. L. Pham, C. Xu and J. L. Prince, "A Survey of Current Methods in Medical Image Segmentation," Annual Review of Biomedical Engineering, 1998

[48] K. V. Asari, "A fast and accurate segmentation technique for the extraction of gastrointestinal lumen from endoscopic images," Medical Engineering & Physics, Volume 22, Issue 2, Pages 89-96, March 2000.

[49] U. Bockholt, W. Muller-Wittig, M. Chrupcala, H. Wang, G. Voss and A. Bisler, "Classification and Segmentation of Malign Bladder Tissue in Endoscopic Images via Statistical Texture Analysis," CG topics 5, 2003.

[50] S. A. Karkanis, D. K. Iakovidis, D. E. Maroulis, D. A. Karras and M. D. Tzivras "Computer Aided Tumor Detection in Endoscopic Video using Color Wavelet Features," IEEE Transactions on Information Technology in Biomedicine, vol. 7, no. 3, 2003.

[51] D. C. Marr and E. Hildreth, "Theory of edge detection," Proc. Roy. Soc. London, B, vol. B207, pp. 187-217, 1983.

[52] B. K. P. Horn and B. G. Schunck, "Determining Optical Flow," Artificial Intelligence, 17, pp. 185-203, 1981.

[53] B. Lucas and T. Kanade, "An Iterative Image Registration Technique with an Application to Stereo Vision," Proc. of 7th International Joint Conference on Artificial Intelligence (IJCAI), pp. 674-679.

[54] M. J. Black and P. Anandan, "The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields," Computer Vision and Image Understanding, CVIU, 63(1), pp. 75-104, Jan. 1996.

[55] D. Bourgin, Color space FAQ, newsgroup,
http://www.neuro.sfc.keio.ac.jp/~aly/polygon/info/color-space-faq.html.

[56] J. Roubert, "Automatic Guidance of a Laparoscope Using Computer Vision," master thesis, Lund University, November 2002.

[57] N. Christofides, Graph Theory: An Algorithmic Approach. New York: Academic, 1975.

[58] O. J. Morris, M. J. Lee and A. G. Constantinides, "Graph theory for image analysis: An approach based on the shortest spanning tree," in Proc. Inst. Elect. Eng., vol. 133, April 1986, pp. 146–152.

[59] S. H. Kwok, A. G. Constantinides and W.-C. Siu, "An Efficient Recursive Shortest Spanning Tree Algorithm Using Linking Properties," IEEE Transactions on Circuits And Systems for Video Technology, Vol. 14, No. 6, June 2004.

[60] J. Shi and J. Malik, "Normalized Cuts and Image Segmentation," IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22, No. 8, pp. 888-905, 2000.

[61] J. Mulroy, "Video content extraction: Review of current automatic segmentation algorithms," Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS97), June 1997.

[62] S. Cooray, N. O'Connor, S. Marlow, N. Murphy and T. Curran, "Semi-Automatic Video Object Segmentation using Recursive Shortest Spanning Tree and Binary Partition Tree," Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS 2001), May 2001.

[63] R. Piroddi and T. Vlachos, "Object-based Segmentation of Moving Sequences using Multiple Features," IEEE Proc. Digital Signal Processing, July 2002.

[64] E. Tuncel and L. Onural, "Utilization of the Recursive Shortest Spanning Tree Algorithm for Video-Object Segmentation by 2-D Affine Motion Modeling", IEEE transactions on circuits and systems for video technology, vol. 10, No. 5, August 2000.

[65] T. H. Cormen, C. E. Leiserson and R. L. Rivest, Introduction to Algorithms, MIT Press, McGraw-Hill, 1990.

[66] T. Brox, D. Farin and P.H.N. de With, "Multi-stage region merging for image segmentation," in Proceedings of the 22nd Symposium on Information Theory in the Benelux. May 2001.

[67] J. Xuan, T. Adali and Y. Wang, "Segmentation of Magnetic Resonance Brain Image: Integrating Region Growing and Edge Detection," ICIP-C 95, pp. 544-547.

[68] M. Van Droogenbroeck and H. Talbot, "Segmentation by adaptive prediction and region merging," In Digital Image Computing Techniques and Applications, Volume II, DICTA 2003, pp. 561-570, December 2003.

[69] C. Xu, "Deformable Models with Application to Human Cerebral Cortex Reconstruction from Magnetic Resonance Images," PhD thesis, Johns Hopkins University, Baltimore, USA, 1999.

[70] Open Source Computer Vision Library Project, website, http://sourceforge.net/projects/opencvlibrary/

[71] C. Xu and J. L. Prince, Active Contours and Gradient Vector Flow, website, http://iacl.ece.jhu.edu/projects/gvf/

[72] D. J. Williams and M. Shah. "A Fast Algorithm for Active Contours and Curvature Estimation," CVGIP: Image Understanding, Vol. 55, No. 1, pp. 14-26, Jan., 1992.

[73] Y. Shu, G. A. Bilodeau and F. Cheriet, "Segmentation of Laparoscopic Images: Integrating Graph-Based Segmentation and Multistage Region Merging", in the 2nd Canadian Conference on Computer and Robot Vision (CRV 2005), May 2005.

[74] W. D. Streidt, Brightness/Contrast Code, newsgroup, http://visca.com/ffactory/archives/5-99/msg00021.html

Appendix I Image segmentation methods

Image segmentation is the partition of an image into a set of non-overlapping regions whose union is the entire image. The purpose of image segmentation is to decompose the image into regions that are meaningful with respect to a particular application. The regions can then be tracked by making correspondence between frames in the circumstance of a video sequence. It is very difficult to tell what actually constitutes a "meaningful" region. However, general segmentation procedures tend to obey the following rules [7]:

- Each region in the segmentation should be uniform and homogeneous with respect to certain characteristics, such as gray level or texture. Different regions should have significantly different values with respect to the characteristics on which they are uniform.
- Region interiors should be simple and without small holes.
- Boundaries of each segment should be simple, not ragged and must be spatially accurate.

The first rule is the basis of almost all segmentation methods, whereas the other two are not necessary true for some applications. Image segmentation is one of the primary steps in image analysis and object recognition. There are numerous sources in the literature addressing this issue. A short review of the most typical approaches to image segmentation is presented in the following subsections. More details on specific methods can be found in many image processing books [7, 8, 9, 10].

I.1 Histogram thresholding

Thresholding is perhaps the most intuitive form of image segmentation. If a pixel in the image has an intensity value less than a threshold value, it is set to one cluster. Otherwise, if the pixel intensity value is greater than or equal to the threshold intensity, the pixel is set to another cluster. Multiple thresholds can be used to classify the image into several clusters. In histogram-based thresholding, the image histogram is used for setting various thresholds to partition the given image into distinct regions.

Definition: The histogram h(i) of an image f(x,y) is a function of the permissible intensity values. In a typical imaging system, intensity takes on values between 0 (black) and 255 (white). A graph that shows, for each gray level, the number of times that level occurs in the image is called the histogram of an image. [11]

It is expected that each region within the image will have some mean grey level intensity and a small spread around this central value. By examining the various peaks of the histogram (denoting grey levels that occur with the highest frequency), we can use them as thresholds to partition the image.

In the easiest case, in which the histogram of the image is bimodal, one simply needs to examine the histogram and to place the threshold in the valley between the two modes. Unfortunately histograms are seldom simple as that and some additional processing is generally needed. In order to reduce the influence of noise and variation of illumination, local or adaptive thresholding techniques have been proposed in [12]. In [13], a technique was developed to find the global minimum of a function of several variables, which even works for functions that have more than one minimum.

In addition to gray level images, thresholding methods have also been applied to multidimensional data. Li et al. [14] propose that the use of two-dimensional histograms of an image is more useful for finding thresholds for segmentation rather than just using gray level information in one dimension. In 2D histograms, the gray level of each pixel as well as the local gray level average of its neighbourhood is used. The authors showed that the application of Fisher linear discriminant to the histogram results in an optimal projection where the data clusters are better defined and hence easier to separate by choosing appropriate thresholds. The experimental results showed that the technique has the same computational demands as one-dimensional techniques but gives better segmentation results.

I.2 Edge-based segmentation

Edges are pixels where an image function (e.g. brightness) changes abruptly. A change of the image function can be described by a gradient that points in the direction of the largest growth of the image function. The gradient magnitude and gradient direction are continuous image functions calculated as:

$$|\, grad\ I(x,y) \,| = \sqrt{(\frac{\delta I}{\delta x})^2 + (\frac{\delta I}{\delta y})^2} \qquad \text{(Equation I.1)}$$

$$\psi = \arg(\frac{\delta I}{\delta x}, \frac{\delta I}{\delta y}) \qquad \text{(Equation I.2)}$$

where arg(x,y) is the angle (in radians) from the x-axis to the point (x, y).

A digital image is discrete in nature, so derivatives must be approximated by differences. The first differences of image intensity in the vertical direction (for fixed i) and in the horizontal direction (for fixed j) are given by

$$\begin{aligned} \Delta_i I(i,j) &= I(i,j) - I(i-n,j) \\ \Delta_j I(i,j) &= I(i,j) - I(i,j-n) \end{aligned} \qquad \text{(Equation I.3)}$$

where n is a small integer, usually 1. The value n should be chosen small enough to provide a good approximation to the derivative, but large enough to neglect unimportant changes in the intensity.

Several often-used first order operators for edge detection were introduced (among others) by Roberts, Prewitt and Sobel [8]. The gradient is approximated in digital images by a convolution with these operators. The Sobel operator uses two kernels oriented in the row and column directions (see Figure I.1). Letting x and y be the values calculated in the two directions, the gradient magnitude is computed as square root of the sum of x^2 and y^2, while the gradient direction is expressed by arctan(y/x).

-1	0	1
-2	0	2
-1	0	1

x

-1	-2	-1
0	0	0
1	2	1

y

Figure I. 1 Sobel edge detector kernels

Edge detection techniques like the Kirsch, Sobel, Prewitt operators are based on convolution in very small neighborhoods and work well for specific images only. The main disadvantage of these edge detectors is their dependence on the size of objects and kernels, and sensitivity to noise. Other edge detection techniques are based on the zero crossings of the second derivative. Like the methods that use the first derivative, they rely on the fact that a step edge corresponds to an abrupt change in the image function. The first derivative of the image function should have an extreme at the position corresponding to the edge in the image, and so the second derivative should be zero at the same position. It is much easier and more precise to find a zero crossing position than an extreme. An example of utilizing the zero crossings of the second derivative is the Marr-Hildreth operator [51]. Note that all these methods do not use spatial information about other edges to allow the detection of nearby weaker edges.

The Canny operator [16] was designed to be an optimal edge detector (according to particular criteria – there are other detectors around that also claim to be optimal with respect to slightly different criteria), and addresses the drawbacks of basic edge detector. It takes as input a grey scale image, and produces as output an image showing the positions of tracked intensity discontinuities. The Canny operator works in a multi-stage process. First of all the image is smoothed by Gaussian convolution. Then a simple 2-D first derivative operator (somewhat like the Roberts Cross) is applied to the smoothed image to highlight regions of the image with high first spatial derivatives. Edges give rise to ridges in the gradient magnitude image. The algorithm then tracks along the top of these ridges and sets to zero all pixels that are not actually on the ridge top so as to give a thin line in the output, a process known as non-maximal suppression. The tracking process exhibits hysteresis controlled by two thresholds: T1 and T2 with T1 > T2. Tracking can only begin at a point on a ridge higher than T1. Tracking then continues in both directions out from that point until the height of the ridge falls below T2. This hysteresis helps to ensure that noisy edges are not broken up into multiple edge fragments.

I.3 Region based segmentation methods

Region based segmentation methods are generally better in noisy images where edges are difficult to detect, because noisy pixels giving rise to spurious edges can be integrated into regions based on the properties of the neighbouring pixels. Region growing algorithm is one of the simplest and most popular algorithms for region based segmentation. It is based on the growth of homogeneous regions according to certain features as intensity, color or texture. The most traditional implementation starts by choosing a starting point called seed pixel. Then, the region grows by adding similar neighbouring pixels according to a certain homogeneity criterion, increasing step by step the size of the region. Thus, the homogeneity criterion has the function of deciding whether a pixel belongs to the growing region or not. The decision of merging is generally taken based only on the contrast between the evaluated pixel and the region.

Graph theory is frequently referenced in region based segmentation methods. It regards each pixel as a node in a graph. Neighbouring pixels whose properties are similar enough are joined by an arc. The definition of "similar enough" can follow different approaches: for instance, Bryant [17] normalizes the difference of the grey level with the standard deviation of the pixel difference over the entire image. The image segments are maximal sets of pixels all belonging to the same connected component. In recursive shortest spanning tree (RSST) based methods [18], a pixel value is compared with the mean of an existing (yet not necessarily complete) neighbouring segment. The pixel is added to the neighbouring segment whose mean value is close to that of the pixel itself. The mean value of the segment is consequently updated. In this work, we used an efficient variant of RSST algorithm [5]. We selected this method because it allows constructing small uniform regions efficiently, without erroneously merging different regions that might not belong together considering higher level knowledge. We then refine the segmentation with our proposed multistage region merging. The graph-based segmentation method will be further introduced in section 1.4.

Region splitting is the opposite of region merging. It begins with the whole image represented as a single region that does not usually satisfy the condition of homogeneity.

The existing image regions are sequentially split to satisfy the given criteria of homogeneity. Region splitting methods generally use similar criteria of homogeneity as region merging methods, and only differ in the direction of their application. A combination of splitting and merging may result in a method with the advantages of both approaches.

Typical split and merge techniques consist of two basic steps. First, the whole image is considered as one region. If this region does not satisfy a homogeneity criterion the region is split into four quadrants (subregions) and each quadrant is tested in the same way; this process is recursively repeated until every square region created in this way contains homogeneous pixels. Next, in the second step, all adjacent regions with similar attributes may be merged following other (or the same) criteria. The criterion of homogeneity is generally based on the analysis of the chromatic characteristics of the region. In general the choice of the splitting and merging criteria are critical for the final segmentation results. Suk and Chung [19] propose a technique in which overlapped 2 x 2 subregions are initially tested against 12 connectivity partition schemes for the four pixels involved. In a second pass, this connectivity information is used in order to merge the subregions to obtain the final segmentation.

I.4 Clustering-based segmentation

One natural view of segmentation is to determine which components of a data set naturally "belong together" regardless of their location. This is a problem known as clustering. Generally, clustering is done in the way of partitioning or grouping. There are two basic algorithms for clustering. In divisive clustering, the entire data set is regarded as a cluster and clusters are recursively split to yield a good clustering. In agglomerative clustering, each data item is regarded as a cluster and clusters are recursively merged to yield a good clustering.

Clustering is done in feature space. It is based on the following elements. Let $X = \{x_1, \ldots x_n\}$ be the data set, $M = \{\mu_1, \ldots \mu_c\}$ the set of centroids corresponding to the c

classes, $\|\bullet\|^2$ is a distance measure in the feature space and $U = u_{ik}$ is a matrix of belongingness of dimensions $n \times c$ such that:

$$u_{ik} = \begin{cases} 1 & \text{if element } x_i \text{ belongs to cluster k} \\ 0 & \text{otherwise} \end{cases}$$

with $\sum_{i=1}^{c} u_{ik} = 1 \ \forall k$, which guarantees that each element belongs to one and only one cluster. A clustering algorithm can be defined as a scheme that evaluates a matrix U to minimize an objective function:

$$J(c, X, U, M) = \sum_{k=1}^{n} \sum_{i=1}^{c} u_{ik} \left\| \mu_i - x_k \right\|^2 \qquad \text{(Equation I.4)}$$

This is often referred to as c-means algorithm. When non-integer values are allowed for matrix U, i.e., $u_{ik} \in [0,1]$, this formulation corresponds to the case of fuzzy c-means algorithm.

Some issues related to clustering are worth studying including how many clusters are the best and how to determine the validity of clusters. In some clustering techniques, such as fuzzy c-means clustering, the number of clusters present in the image has to be specified in advance. Also, clustering algorithms do not directly incorporate spatial modeling and can therefore be sensitive to noise and intensity inhomogeneities. Frigui and Krishnapuram have addressed three major issues associated with conventional partitional clustering [20]. The issues addressed are sensitivity to initialization, difficulty in determining the number of clusters and sensitivity to noise and outliers.

I.5 Segmentation by statistical methods

Statistical segmentation methods utilize the techniques developed in the field of statistical pattern recognition to classify pixels into different regions. In statistical pattern recognition [28], a pattern is represented by a set of d features, or attributes, viewed as a d-dimensional feature vector. The recognition system works in two modes: training/learning and classification/testing (see Figure I.2). The role of the pre-processing module is to remove noise, normalize the pattern, and any other operation that will

contribute in defining a compact representation of the pattern. In the training mode, the feature extraction/selection module finds the appropriate features for representing the input patterns and the classifier is trained to partition the feature space. The feedback path allows a designer to optimize the pre-processing and feature selection strategies. In the classification mode, the trained classifier assigns the input pattern to one of the pattern classes based on the measured features.

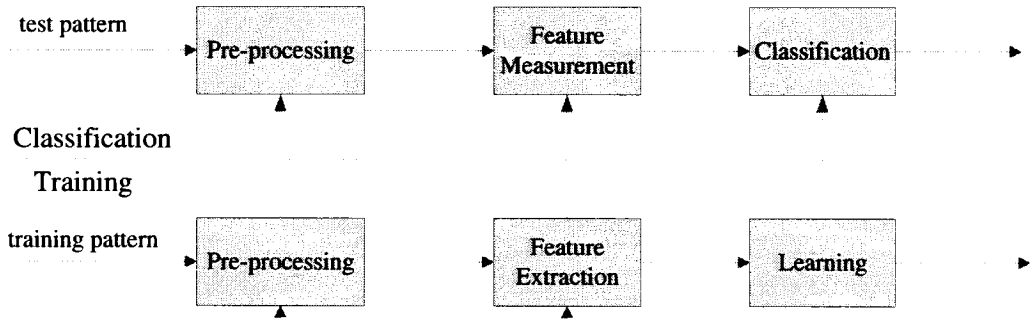


Figure I. 2 Statistical pattern recognition model

The decision making process can be summarized as follows. A given pattern is to be assigned to one of c categories/classes $\omega_1, \omega_2, \ldots, \omega_c$ based on a feature vector of d values $\boldsymbol{x} = (x_1, x_2, \ldots, x_d)$. The features are assumed to have a probability density function conditioned on the pattern class. Thus, a pattern vector $\boldsymbol{x}$ belonging to class ω_i is viewed as an observation drawn randomly from the class-conditional probability function $p(\boldsymbol{x}|\omega_i)$. A number of decision rules such as Bayes decision rule and maximum likelihood rule, are available to define the decision boundary. The "optimal" Bayes decision rule for minimizing the risk (expected value of the loss function) can be stated as follows: Assign input pattern $\boldsymbol{x}$ to class ω_i for which the conditional risk

$$R(\omega_i \mid x) = \sum_{j=1}^{c} L(\omega_i, \omega_j) \cdot P(\omega_j \mid x) \qquad \text{(Equation I.5)}$$

is minimum, where $L(\omega_i, \omega_j)$ is the lost function incurred in deciding ω_i when the true class is ω_j and $P(\omega_j|\boldsymbol{x})$ is the posterior probability. In the case of 0/1 loss function, i.e,

$$L(\omega_i, \omega_j) = \begin{cases} 0, & i = j \\ 1, & i \neq j \end{cases}, \qquad \text{(Equation I.6)}$$

the conditional risk becomes the conditional probability of misclassification. For this choice of loss function, the Bayes decision rule can be simplified as follows: Assign input pattern $\boldsymbol{x}$ to class ω_i if $P(\omega_i \mid x) > P(\omega_j \mid x)$ for all $j \neq i$. This is often cited as maximum a posteriori (MAP) rule.

The optimal Bayes decision rule is frequently used in designing the classifier. In practice, the class-conditional densities are usually unknown and must be learned from available training patterns. If the form of the density function is known (e.g. multivariate Gaussian), but some of the parameters (e.g. mean vector and covariant matrix) are unknown, then we have a parametric decision problem. A common strategy for this kind of problem is to substitute the unknown parameters with their estimated values (learned from training).

Appendix II Description of motion information

II.1 Global motion approximation

In video compression it is desirable to model global motion using a parametric model so that the parameters may be used to fully recreate the approximation during decoding, whilst only requiring the transmission of a few bits. Parametric models express a transformation, mapping pixels in the reference frame $p = (x, y)$ to those in the current frame $p' = (x', y')$. This can be written as $p' = p + v$ where v is the motion vector. Different models for the global motion vector field G vary in complexity and ability, some of which are given in Table II.1 below.

Model	Transformation	Details
Translation	$\boldsymbol{G = b}$	$\boldsymbol{b} = \begin{pmatrix} b_x \\ b_y \end{pmatrix}, \boldsymbol{A} = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}, \boldsymbol{C} = \begin{pmatrix} c_x \\ c_y \end{pmatrix}$
Scaling and translation	$\boldsymbol{G} = \mathrm{k}\boldsymbol{p} + \boldsymbol{b}$	
Affine	$\boldsymbol{G = Ap + b}$	
Projective	$\boldsymbol{G} = (\boldsymbol{Ap} + \boldsymbol{b}) / (\boldsymbol{C}^T\boldsymbol{p} + 1)$	

Table II.1 Parametric motion models

The most simple translation model consists of 2 parameters and may only model a global shift. The next more complex model also captures isotropic scaling (equal in both x and y directions) and requires 3 parameters. The affine model is more general and can model anisotropic zooming, rotation and pure shear along with simple translation. Six parameters are required for affine model: 2 translation parameters and a 2 x 2 matrix that describes the more complex modes of movement. The projective model requires 8 parameters and is the most flexible one. It can be used to describe any form of 3D translation and rotation, for example that of a change in camera position in 3D space. To simplify the computation, the affine model is often written as

$$\begin{aligned} x' &= k_1 x + k_2 y + k_3 \\ y' &= k_3 x + k_4 y + k_6 \end{aligned}, \qquad \text{(Equation II.1)}$$

and the projective model is approximated by the bilinear model

$$\begin{aligned} x' &= k_1 xy + k_2 x + k_3 y + k_4 \\ y' &= k_5 xy + k_6 x + k_7 y + k_8 \end{aligned} \quad \text{(Equation II.2)}$$

As far as the description of the motion is concerned, the situation of a fixed camera with a moving scene is equivalent to that of a moving camera and a fixed scene. However, when multiple objects are present in the scene, it can be useful to separate the contribution to the motion given by the objects and the one due to the camera motion. Particularly when motion estimation is performed for segmentation purposes, some authors propose to extract from the scene a parametric description (typically in the form of Equation II.1 or Equation II.2) of the camera motion, and to eliminate the contribution of global motion. The residual motion can then be attributed to the movement of the objects themselves. This form of pre-processing of the motion information is performed for example in [22, 25].

II.2 Optical flow

Assume $I(x, y, t)$ is the center pixel in a $n \times n$ neighbourhood and moves by δx, δy in time δt to $I(x + \delta x, y + \delta y, t + \delta t)$. Since $I(x, y, t)$ and $I(x + \delta x, y + \delta y, t + \delta t)$ are the images of the same point (and therefore the same) we have:

$$I(x, y, t) = I(x + \delta x, y + \delta y, t + \delta t) \quad \text{(Equation II.3)}$$

This assumption forms the basis of the 2D Motion Constraint Equation. The assumption is true to a first approximation (small local translations) provided δx, δy, δt are not too big. We can perform a first order Taylor series expansion about $I(x, y, t)$ in Equation II.3 to obtain:

$$I(x+\delta x, y+\delta y, t+\delta t) = I(x, y, t) + \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t + \text{H.O.T.} \quad \text{(Equation II.4)}$$

where H.O.T. are the higher order terms. Using the above two equations we obtain $I_x u + I_y v + I_t \approx 0$, in which (u, v) are the x and y components of optical flow and (I_x, I_y, I_t) are the partial intensity derivatives with respect to x, y and t. One way to solve the optical flow is to use least-squares calculation. Let $e = I_x u + I_y v + I_t$, we want to minimize the error, e^2, with respect to (u, v) over a region R.

$$\frac{\partial}{\partial u}\sum_{R} e^2 = 2\sum_{R} e I_x = 0$$
$$\frac{\partial}{\partial v}\sum_{R} e^2 = 2\sum_{R} e I_y = 0 \qquad \text{(Equation II.5)}$$

$$\begin{pmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{pmatrix}\begin{pmatrix} u \\ v \end{pmatrix} = -\begin{pmatrix} \sum I_t I_x \\ \sum I_t I_y \end{pmatrix} \qquad \text{(Equation II.6)}$$

Equation II.6 can be solved in closed form when $\begin{pmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{pmatrix}$ is a non-singular matrix [23].

Most motion segmentation approaches have to extract motion information from pixel values of video frames. This is done on raw images or decoded/reconstructed images though certain complex analysis on two or more consecutive frames. It is worth mentioning that some kind of motion information has already been included in video streams encoded by widely accepted video format standards. For example, the MPEG standard serials adopted a technique called motion estimation/compensation, which calculates the displacements of consecutive matching blocks as motion vectors and encode into standard MPEG streams. In many cases, especially well textured objects, the motion vector values reflect the movement of objects in the scene very well. So instead of having a separate module to extract motion information, some approaches utilized these motion vector values directly [36, 37]. Although this could reduce the accuracy of object boundary because motion vectors are associated only with block of pixels, it does not result in a major problem in many applications because the goal of tracking is not to determine the exact correspondence for every image location in a pair of images, but rather to determine, in a global sense, the movement of an entire target region over a long sequence of images.